Portfolio Overview

Reconfigurable logic test methodologies for ISDN.

Patrick Sugrue

This portfolio is comprised of three projects and is submitted in partial fulfilment of the requirements of the University of Glamorgan for the degree of Master of Philosophy.

School of Electronics

University of Glamorgan

Pontypridd

CF37 1DL

15 – December – 2003

Certificate of research

This is to certify that, except where specific reference is made, the work described in this portfolio is the result of the candidate. Neither this portfolio, nor any part of it, has been presented, or is currently submitted, in candidature for any degree at any other University.

Signed ..... Patrick Sugrue .....

Candidate.

Signed ..... M. Wahab .....

Director of Studies.

Date ..... 15 - 12 - 2003 .....

Acknowledgements

I would like to sincerely thank my director of studies Prof M.A. Wahab for his support and technical guidance during this work. His advice during the writing of this portfolio was invaluable. I would also like to thank Mr Clive Thomas for the extensive support received from the Centre for Electronic Product Engineering that made this portfolio possible to complete. I would like to extend my gratitude to Dr Jurgen Richter and Dr Ralf Patz for their technical advice and comments on various aspects of the work.

I would like to acknowledge the support of the technical staff at Chesilvale Electronics Ltd.

Finally, I would like to thank my family for their encouragement and support over the past few years.

Contents

List of figures.

Glossary.

Symbol.	Description.
$x(n)$	A discrete data sequence where n is the index of each sample.
$PE(\omega)$	The Power Spectrum Density
$C_{xx}(m)$	The Autocovariance function of a discrete data sequence X(n).
$X(k)$	The Fourier transform
k	The harmonic number of the transform component.
f_s	The Sampling Frequency.
Δf	The frequency Resolution or Frequency Bin.

N	Discrete sample length.
n	Sample index.
$\lvert X(k)\rvert^2$	The power spectrum density.
$\tilde{X}(K)$	The Complex Conjugate.
$w_B(m)$	The Bartlett Window.

Abbreviations.

ADSL	Asymmetric digital subscriber line.
ANSI	American National Standards Institute.
ASSP	Application specific standard product.
ASIC	Application specific integrated circuits.
ATIS	Alliance for Telecommunications Industry Solutions.
AWG	American wire gauge.
BERT	Bit error rate test.
CCITT	International Telegraph and Telephone Consultative Committee.
CPLD	Complex programmable logic device.
dBm	Decibels referenced to one mili- watt.
DCO	Digitally controlled oscillator.
DFT	Discrete fourier transform.
DPLL	Digital phase locked loop.
DUT	Device under test.
DSL	Digital subscriber line.
ECSA	Exchange Carrier Standards Association.

ENBW	Equivalent noise bandwidth.
FEXT	Far-end crosstalk.
FFT	Fast fourier transform.
FPGA	Field programmable gate array.
HDLC	Higher-level data link control.
HDSL	High bit-rate digital subscriber line.
ID	Increment-decrement.
IDN	Integrated digital network.
IOM	ISDN orientated modular interface.
ISDN	The Integrated Systems Digital Network.
ITU-T	Telecommunication Standardisation sector of the International Telecommunications Union.
kbps	kilo-bits per second.
LAPD	Link access protocol for the D channel.
LFSR	Linear feedback shift-register.
LT	Line terminator.
mbps	mega-bits per second.
NEXT	Near-end crosstalk.
NT	Network terminator.
OSI	The open systems interconnect.
PCB	Printed circuit board.
PD	Phase detector.
PL	Processing loss.
PLL	Phase locked loop.
PRBS	Pseudo random binary sequence.

PSD	Power spectrum density.
S Bus	Subscriber bus.
SRAM	Static random access memory.
TE	Terminal equipment.
UK	United Kingdom.
VHDL	Very-high-speed hardware description language.
VLSI	Very large scale integration.
2B1Q	Two binary one quaternary.
4B3T	Four binary three ternary.

Abstract

This portfolio presents three separate design and development projects. Each project addresses a different aspect of ISDN testing. Field programmable gate array (FPGA) technology was employed to facilitate the development. The design and development work produced a hardware platform that could be reconfigured to provide the test features associated with each project. The three projects were, a power spectrum density (PSD) measurement facility for basic-rate ISDN, a bit-error rate test (BERT) facility for basic-rate ISDN, and a passive monitor for basic-rate ISDN. The project work demonstrated how greater test feature integration could be achieved while allowing future features to be easily incorporated as software upgrades reducing the development time to market for new products.

1. Introduction.

Digital Subscriber Line (DSL) technology has advanced rapidly over the past two decades. The driving force behind its advancement being the huge domestic demand for fast Internet access along with the business demands for a more integrated digital service. Although challenged in the early nineties by the vastly superior bandwidth capabilities of fiber optics, the cost of fiber installation to transport digital data proved to be too expensive in comparison with the ubiquitous copper telephone lines already in existence. Hence a number of different DSL technologies that offered high data transfer rates over a twisted pair copper telephone line emerged. The acronym xDSL refers in general terms to this technology and the x character is replaced with a specific letter to denote the DSL service in question. The Integrated Systems Digital Network (ISDN) is also a DSL but is generally referred to as simply ISDN rather than ISDN-DSL. This portfolio concentrates on the theme of reconfigurable test methodologies for Basic Rate ISDN. The Portfolio is composed of three projects. Each project focuses on the improvement of current methods used to facilitate testing at the ISDN physical layer. This overview chapter presents a thematic background that links the three projects, the literature survey and a summary of the individual projects.

1.1. The problems with currently deployed ISDN test equipment.

There are a number of problems with current ISDN test equipment designs. Companies involved in the development of ISDN test equipment have traditionally used Application Specific Standard Products (ASSP) as core components to realise their designs [39], [40], [41]. The adoption of commercially available ASSP chip sets effectively means design ownership of the key components belongs to another organisation. This dependence on ASSP chip sets to realise designs also reduces design flexibility. The physical layer features currently provided by ISDN test equipment are those associated with currently available chip sets. Such a design change may be required if an ISDN test equipment designer requires a feature not supported by currently available chip sets. An ASSP design change is costly and time consuming. Test equipment designers may not be successful in demanding specific features from chip manufacturers due to the low volumes associated with the test equipment market.

Different test equipment is used in the installation and maintenance of each of the currently deployed DSL technologies. As the DSL technology advances new equipment with new features is required to address the changes. Technicians and engineers involved in the development of ISDN equipment and the installation and testing of ISDN networks currently require different test equipment to test different aspects of ISDN. It is common to find that ISDN technicians require more than one piece of test equipment to identify a network fault. ISDN is therefore, one area where the adoption of reconfigurable logic technology can be used to incorporate different test capabilities within a single piece of equipment and hence, reduce cost.

Reconfigurable technology has not been effectively exploited by the ISDN industry to date.

Some improvements have already been made in the field of computer networks through the employment of reconfigurable logic. Taylor et al demonstrated how reconfigurable logic technology was utilised to improve router designs [1]. Stefanopoulos et al discussed how FPGA technology can be utilised to provide a reconfigurable hardware platform for basic rate and primary rate ISDN lines [2]. The paper describes the use of the FPGA to provide an ISDN data buffering facility. It was also shown how the FPGA could be reconfigured to provide Interface control logic for different ISDN ASSPs on the same board. However, the implementation method presented still relied on the use of ASSPs to provide ISDN transceiver functionality. To date, little work has been done to advance the area of reconfigurable test methodologies for ISDN. The vast integration capabilities of currently available FPGAs have not been considered to realise ISDN transceiver implementations. Considering the dynamic nature of the DSL market, the development of a reconfigurable hardware platform that can be upgraded, as new features are demanded and as the DSL technology advances will be an ideal solution.

1.2. Background.

The move towards a digital telephone network began in the 1960s when analogue data representing speech was digitised at the local exchange and transmitted over the trunk network. Telephone switching on the other hand used predominately analogue signals and this situation remained until the early 1980s when the introduction of system X signified the first step towards achieving an integrated

digital network in the United Kingdom [3]. With system X, data rate for a single telephone call was 64 kbps and is derived as follows. Human speech has a maximum frequency of 4kHz. According to Nyquist theory, if an analogue signal is sampled at twice the frequency of the highest frequency component in the signal then the sampled data will contain all the information of the original signal. In digital telecommunications analogue speech signals are therefore sampled at 8kHz and are represented by a 12 bit digital number. It is possible to tolerate lower accuracy at higher speech levels and a logarithmic scale may be deduced to allow compression of the sampled 12 bit speech signal into an 8 bit number. Two techniques were used. A Law is used in Europe and μ Law is used in the USA. A Law and μ Law were discussed in detail by Dunlop and Smith [3]. Therefore, the channel capacity required to transmit a speech call is 64 kbps (8kHz multiplied by 8 bits).

At the local exchange incoming telephone calls were time division multiplexed into thirty 64kbps channels for onward transmission. A further two 64kbps channels were used for synchronisation and signaling purposes. This produced 32 channels with a data rate of 2.048mbps. System X was extensively deployed and over the space of a decade all the local exchanges were upgraded to support the transfer of digital data and signaling over the local network. The 2.048mbps data rate is referred to as E1 and is deployed in Europe. The USA and Japan developed their own digital networks based on the same rational and use 24 channels giving a data rate of 1.536mbps. This data rate is referred to as T1.

This integrated the data and signaling facilities offered by local exchanges and created the integrated digital network (IDN) on which subsequent digital services were based. However, the connection between this IDN and the subscriber still used analogue signals. The copper telephone lines that were previously only required to

transport analogue voice signals were not intended to operate at transmission rates greater than 4kHz and therefore, the bandwidth capabilities were questioned. This demanded a review of the signaling technique to facilitate the transmission for higher data rates between the subscriber and the local exchange. The objective of providing a DSL service was to do so without significant modification of the existing telecommunications infrastructure. Therefore, the adopted telephone line signaling that could achieve the required DSL performance were chosen with the existing infrastructure in mind.

Standards and technical specifications were required and study groups comprised of industrial, academic and government bodies were formed to consider all the issues. The Telecommunication Standardisation sector of the International Telecommunications Union (ITU-T), formally known as the International Telegraph and Telephone Consultative Committee (CCITT) was instrumental in producing the DSL recommendations that evolved over the past twenty years [25], [28], [30], [32]. The function of the ITU-T is to provide global telecommunications standards on the technical, operating and tariff issues. The Alliance for Telecommunications Industry Solutions (ATIS) formally called the Exchange Carrier Standards Association (ECSA) and the American National Standards Institute (ANSI) provided the industrial cooperation to create network interconnection standards in the USA [4]. The standards and recommendations that evolved defined the requirements that the telephone lines (loop plant) between the subscriber and the central office should meet before a DSL service could be provided.

The Open Systems Interconnect (OSI) model was used to categorise signals flowing within the DSL system into one of seven layers depending on the signals function [3]. For ISDN systems, only layers one to three are applicable. The standards

recognised that certain imperfections in the loop plant inherited from the analogue era would have to be tolerated by DSL equipment. With the highly developed state of Very Large Scale Integration (VLSI) technology, the ability to design and produce the required complex chip sets to advance the DSL equipment production program is not a barrier. It was clear to telephone companies that the main obstacle to achieving widespread deployment of a high quality SDL service was the suitability of the telephone infrastructure itself.

The telephone infrastructure connecting the subscribers to the local exchange has as its main component a multitude of copper wires used as transmission lines. The transmission lines are manufactured such that two copper wires are tightly twisted together and thus the name twisted pair is often used in reference. Bundles of 50 twisted pairs comprise a single cable and up to two thousand bundles can connect a central office to a distribution point within a subscriber area. Usually these cables are located in close proximity within underground ducting systems. Differential signaling is employed in an attempt to eliminate common mode noise on these cables. A differential signal is comprised of two equal but complementary parts. At the transmitter end each part of the signal is applied to one of the wires in a pair. The existence of noise on the telephone line will affect both wires in the same way. The induced interference will appear as a voltage whose polarity is the same on both wires. At the receiver end a differential amplifier is employed, the common mode noise is cancelled and the original signal is recovered minus the noise.

The current carrying capability of the wire is dependant on its thickness. The thickness is quantified using American Wire Gauge (AWG). The loop plant will often employ wires of different thickneses to complete a connection between a subscriber and the central office. It is common to encounter wire gauges of 22 AWG (0.34 mm^2),

24 AWG (0.25 mm^2), and 26 AWG (0.14 mm^2) all used on a single loop. The technique used to join different wires is called a splice. Another potential impairment that can affect DSL deployment is the bridge tap. A bridge tap is a disused twisted pair that is still connected to the loop but which may not be properly terminated at its end. Bridge taps may run for miles and the problem is especially widespread in the non-densely populated areas of North America.

Reconditioning the loop plant in these areas is an expensive exercise. If DSL technology is to be deployed over these loops then its implementation must tolerate loop imperfections and still function as expected. In the case of basic rate ISDN the ITU-T recommendation T1.601, specifies fifteen loops that should be capable of supporting ISDN data rates [4]. Some of the loops specified show mixtures of different wire gauge and bridge tap arrangements. The loop plant already described is called the non-loaded loop plant. The loaded loop plant contains inductors referred to as loading coils. Loading coils filter out high frequency noise above 4kHz and help to improve voice quality in loops greater than 18,000 feet. European loop plant is virtually free of loading coils. Engineering practice in North America is to load loops longer than 18,000 feet. As DSL services operate at data rates above 4kHz a loaded loop would prohibit deployment. The bandwidth of a non-loaded twisted copper pair can be up to 10MHz. It is the imperfections presented by the existing non-loaded loop plant that can adversely affect the quality of a DSL service and can contribute to reduced bandwidth.

When a telephone company targets a subscriber area for DSL deployment, the suitability of the telephone infrastructure is questioned. Telephone line reconditioning to support DSL data rates is an expensive venture and extensive research and testing is required before DSL deployment is deemed viable. This is usually achieved using

automatic pair qualification tools that are installed in the central office serving the subscriber area in question. Galli and Waring suggested a method to identify the loop makeup using single ended measurements and thus, provide information on which to base deployment decisions [5].

The first digital subscriber line service to gain widespread deployment was ISDN. High Bit-Rate Digital Subscriber Line (HDSL) followed in the early 1990s. HDSL operates using two twisted pairs. Data is transmitted on one twisted pair and received on the other. HDSL was developed as an alternative method to provide a T1 and E1 service. There is no standard for HDSL even though it is one of the most widely deployed DSL services to date. HDSL 2 was developed to provide a fully symmetrical T1 and E1 service over one twisted pair. The ANSI T1E1.4 working group is currently working on the standard for HDSL 2. The main application of HDSL 2 is high-speed data transfer and video conferencing. The high demand for fast Internet access in the mid 1990s saw the development of the Asymmetric Digital Subscriber Line (ADSL) technology. ADSL transmits high data rates downstream from the exchange to the subscriber and a lower data from the subscriber to the exchange. ADSL operates over a single twisted pair and can share the line with an analogue telephone. Data rates of between 1 and 2mbps can be achieved over 18,000 feet with some telephone companies reporting 6mbps over 12,000 feet. The upstream data rate can be up to 640kbps. Coiffi, Silverman and Starr have presented a comprehensive study on DSL technologies, and have outlined the different DSL technologies [6].

The Integrated Systems Digital Network (ISDN).

ISDN combines all the business demands such as voice, data transmission, fax, video and supplementary services into a single package. The package is available as a basic rate service or a primary rate service. The basic rate service offers two bi-directional or fully symmetrical 64kbps channels for transmission of subscriber data and voice, called bearer channels and one 16kbps channel for signaling. Symmetrical or bi-directional transmission is achieved using a hybrid circuit with echo-cancellation. The basic rate service is suitable for small business organisations. The total bandwidth available to the subscriber is 144kbps. A primary rate service offers 30 bearer channels and 2 channels for signaling and maintenance giving a total bit rate of 2.048mbps.

A subscriber requires specific telecommunications equipment to receive an ISDN service. The exchange in the central office also requires ISDN compatible electronics and software. The ISDN network and the reference connection points are shown in figure 1. An ISDN network is composed of terminal equipment (TE) and a network terminator (NT). A TE can be any ISDN compatible device. Within the subscriber's premises a four-wire system called the subscriber bus (S Bus) is used to facilitate data transmission. The TEs in the network are connected via the S bus to the NT. Reference points have been defined to identify connection points in the network [25].

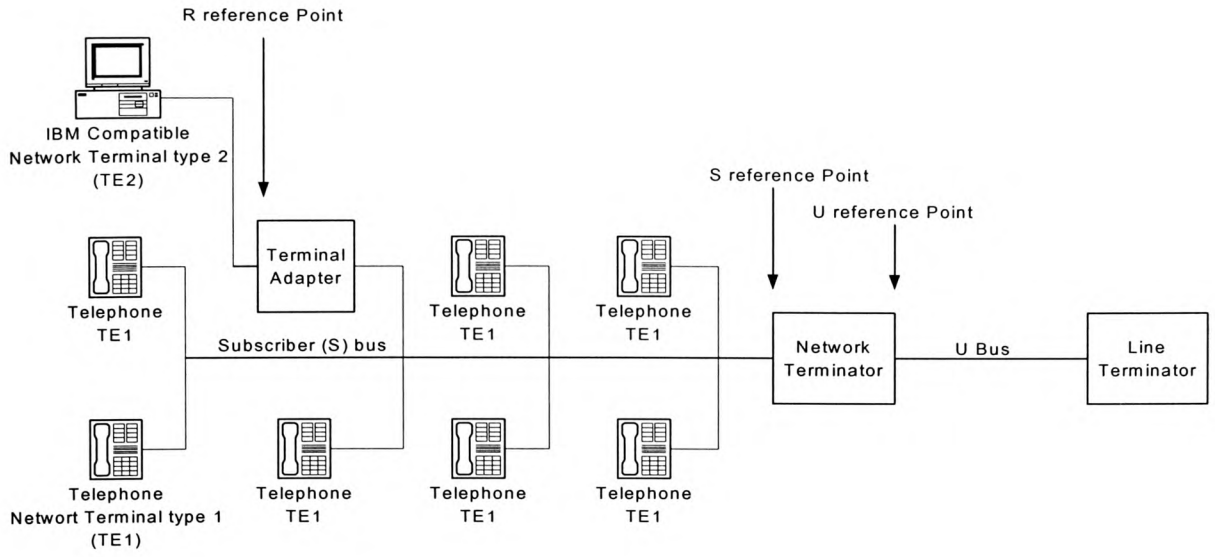


Figure 1 – The ISDN reference points.

The U reference point is where the subscriber's signals enter the telephone companies cabling system. The S reference point, also called the S bus, is where the subscriber connects ISDN compatible equipment. A maximum of eight ISDN compatible TEs may be connected to the S Bus. Non-ISDN equipment is connected to the T reference point and requires a Terminal Adapter before connection to the S Bus is possible. A Network Terminator acts as an interface between the S bus and the U reference point. The U reference point connected to the exchange in the telephone company at the Line Terminator (LT) point via a twisted pair telephone line. In the UK the demarcation point between the subscriber and the telephone company takes place at the S reference point. The telephone company will install the Network Terminator (NT) at the subscriber's premises and the NT remains the property of the telephone company. In the USA the demarcation point between the subscriber and the telephone company is at U reference point. Two types of connections may be set up on a Basic Rate ISDN Network. These are point-to-multi-point and point-to-point connections. If a point-to-point connection type is set up, a single TE may be

connected to an NT. The TE may be located at a distance of 1km from the NT. Alternatively, the subscriber may connect up to eight pieces of Terminal equipment to the S Bus. The ISDN signal from the subscriber enters the telephone companies twisted pair connection at the U reference point. The maximum signal power that can be inserted into the line at the U reference point is 14dBm. If a nominal peak voltage of 2.45V is generated into 135 ohms the output power equates to 13.5dBm.

The Power Spectrum Density.

The power present in the output signal of a basic-rate ISDN transmitter can be represented by a Power Spectrum Density (PSD) plot over a frequency range of 0Hz to 240kHz. A PSD mask defines the signal power limit that can be applied to the ISDN. Transmission power limits are placed on DSL technologies in order to reduce interference with other technologies sharing the same telephone cabling system. The T1E1 committee of the American National Standards Institute (ANSI) has produced the T1E1.4 Draft Spectrum Management Standard where a PSD mask for each of the DSL technologies has been defined [7]. The standard also provides deployment guidelines for telephone companies intending to mix different DSL technologies within the same telephone cabling system. The guidelines exclude the mixing of some DSL technologies within the same telephone cabling due to the resulting electrical interference.

When deploying a DSL technology on a cabling system it is necessary to attain knowledge of the DSL technology already existing on the same cabling system. This can be a difficult task, as the information about telephone wiring is not always well documented. The documentation that exists may not be centralised in one database and often the information available is out of date due to telephone line

modifications that were not recorded. This can result in the inadvertent deployment of a DSL technology that interferes with another DSL technology. Telephone companies employ the use of test equipment with PSD measurement facilities to identify an interfering DSL. When a development engineer is designing an ISDN transmitter circuit the output power is measured using a piece of test equipment with a PSD measurement facility. An engineer requiring a PSD measurement facility is currently forced to purchase a separate measurement instrument dedicated to this task. If an engineer is only interested in the measurement of PSD for ISDN applications then the other features available on the same instrument may not be used.

Once the ISDN network has been installed the most basic commissioning involves assessing the transmission lines Bit Error Rate (BER). This is performed using a test telephone with an ISDN BER test (BERT) feature. The BERT feature for Basic Rate ISDN test applications is a complex piece of equipment. Basic Rate ISDN BER testers include digital electronic features such as S Bus synchronisation and Phase Locked Loop (PLL) circuitry, S Bus maintenance channel synchronisation and control circuitry and the required analogue electronic circuits to facilitate transmission and receiving of S Bus signals. These features are currently integrated within mixed signal S Bus Interface chips. The essential electronic feature required to facilitate a BER test operation is the Pseudo Random Binary Sequence (PRBS) Generator and receiver. Human speech is random in nature and therefore PRBS generators are employed to best simulate speech on telecommunications networks. A PRBS generator may be implemented either in hardware or software. The hardware implementation of PRBS circuits is relatively straightforward. However, none of the currently available S Bus Interface ASSPs have integrated PRBS features.

When a fault occurs on an ISDN network the technician firstly identifies a suspect TE. It is common to substitute the suspect TE with a healthy TE. This allows the technician to ascertain if the problem is due to a faulty TE or NT. To find the exact nature of the fault an ISDN Protocol Monitor is required. A Protocol Monitor allows the technician to see the signals that are flowing between the TE and NT at each layer in the system. Currently available Protocol Monitors also rely on commercially available ASSPs to implement the S Bus interfacing functions. In the case of an ISDN Protocol Monitor, two ASSP chip sets are required. This increases the cost of the equipment.

The general design philosophy of test equipment design engineers is to adopt "off the shelf" solutions in an attempt to increase time to market. The resulting solution is an inflexible and expensive design that has a limited life cycle. When there is a change to a DSL technology or a new technology is conceived, test equipment engineers are forced to wait until the necessary chip sets become available before test equipment development may begin.

Field Programmable Gate Array technology.

Field Programmable Gate Array (FPGA) technology is capable of integrating vast amounts of digital electronic functionality within a single chip [34]. FPGAs are configured with a binary sequence consisting of configuration data to establish logic functionality. Different technologies exist. These are, one time programmable, electrically erasable, and Static Random Access Memory (SRAM) based devices [35], [36], [37]. This project will utilise SRAM based devices in order to provide design flexibility and increased feature integration [38]. During FPGA configuration, the logic functionality is stored within SRAM cells while power is applied to the chip.

The removal of power or a user instigated reconfiguration command wipes the FPGAs memory cells and allows the loading of a different configuration data file thus changing the logic functionality.

1.3. The Portfolio Objectives.

The portfolio objectives are outlined in the following points.

1 To investigate the applicability of programmable logic to the implementation of flexible ISDN test equipment.

2 To develop a flexible ISDN test platform using reconfigurable FPGA devices as the core components to reduce the reliance on ASSPs, shorten the time to market, introduce innovative test features and protect intellectual property.

3 To implement the three main test functions, PSD measurement, Basic Rate ISDN BERT and protocol monitoring, using a configurable logic based universal platform.

4 To investigate the integration of the three test functions listed above in a single test equipment. Three separate projects will be undertaken to address the above objectives and three separate stand-alone reports will be produced. The following points outline the objectives of each project.

5 The main objective of project number 1 is to develop DSP measurement techniques to measure PSD for basic rate ISDN applications.

6 The aim of project number 2 is to develop a BERT feature for basic rate ISDN applications.

7 Project number 3 focuses on the development of an ISDN protocol monitoring feature.

1.4. Structure of Portfolio.

The Portfolio is structured as three individual reports as described below.

Report number 1 (A PSD Measurement Feature for Basic Rate ISDN).

Report number 1 describes the work that was conducted as part of project 1 and covering the hardware and software development to measure the PSD of a 2B1Q line code at the Basic Rate ISDN U reference point. The method to achieve compliance testing against the PSD mask defined in the T1.601 standard is described. The hardware and software is verified and the results are presented. The development process succeeded in producing a reconfigurable solution that can be updated in parallel with the developing the T1E1.4 spectral regulation standard. It is shown how FPGA technology is employed to implement a solution and how new compliance test facilities can be easily incorporated as software upgrades, thus reducing the hardware cost and the time to market.

Report number 2 (A Bit Error Rate Test Feature for Basic Rate ISDN).

Report number 2 presents the design and implementation of hardware components to facilitate BER testing on the basic-rate ISDN. The main design effort concentrates on the circuitry required by the ISDN TE to establish a loopback at the NT, transmit and receive a pseudo-random binary sequence (PRBS) from the NT and count bit errors in the looped back PRBS over the test duration. The analogue circuitry required to interface with the ISDN S interface is also designed and implemented. An NT design is realised with the functionality to facilitate test and validation of the TE. The design work employs a top down synchronous design technique using the VHDL hardware description language. The digital electronics for

both the NT and TE is implemented on a Xilinx FPGA. It was concluded that an FPGA based S Interface ISDN transceiver can be designed, implemented and employed as an alternative to ISDN chip sets.

Report number 3 (An ISDN Protocol Monitor for Basic Rate ISDN).

Report number 3 presents the hardware circuitry required to facilitate ISDN protocol monitoring on the S Bus. It was shown how the Layer one signals are identified and the D channel data is extracted from the data frames flowing between the NT and TE. The report described how a VHDL test bench was employed to simulate the signals on the S Bus. It described how design components developed in project number 2 were employed again in project number 3. It was demonstrated how design ownership coupled with the design flexibility of FPGA technology can allow test equipment designers to respond quickly to market demands with more efficient and economic designs.

2. Literature review.

The following chapter presents the literature review for the three projects.

2.1. Literature review for Project Number 1.

ISDN uses a fully symmetrical or full duplex data transmission system, meaning that data is sent and received simultaneously. Fully symmetrical data transmission can be achieved by four-wire operation, with one pair for transmission and the other for receiving. Frequency separation of both transmit and receive signals is achieved using a hybrid circuit with echo-cancellation.

The ISDN hybrid circuit.

The hybrid circuit used in an ISDN transmission system is required to deal with differential signals and this circuit has been presented in [9]. Figure 2 shows the hybrid symbol. The transmission line signal and both transmitted and received signals are differential signals. The hybrid circuit is composed of four impedance elements arranged as a wheatstone bridge configuration. The characteristic impedance of the transmission line forms one of the impedance elements of the bridge configuration. In an ideal hybrid circuit the characteristic impedance of the transmission line would equal the other impedance elements of the bridge, denoted as Z. In this case, the transmitted signal would pass attenuated by the impedance Z onto the transmission line. In the ideal case the amplitude of the transmitted signal would be the same if measured across each of the separate impedance elements of the bridge. If a difference amplifier is employed and connected to the received signal, then its output voltage would be zero, thus isolating the transmitted signal. If a distant hybrid is to

transmit a signal, a voltage difference will develop across the difference amplifier proportional to the difference between the locally transmitted signal and the distant hybrids transmitted signal.

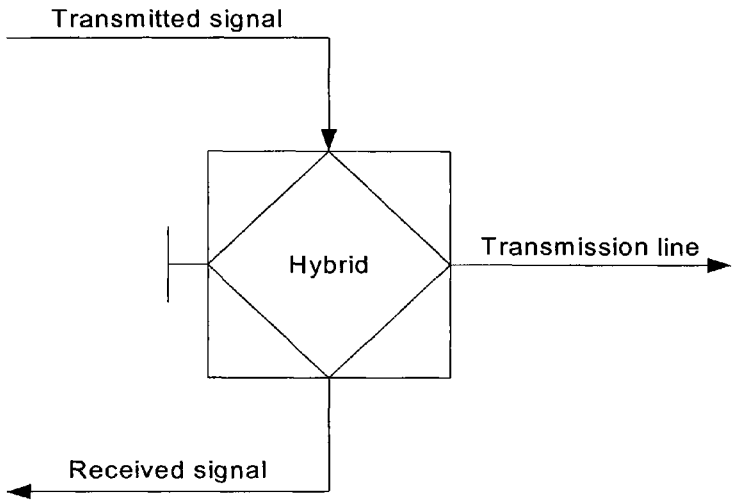


Figure 2 - ISDN Hybrid symbol.

The received signal can now be recovered and separated from the transmitted signal. However, in practice it is difficult to match the impedances precisely. In this case, a voltage difference is developed across the hybrid connections to the difference amplifier. The difference amplifier interprets this error signal as a valid signal. This error voltage is termed echo. To avoid this, the transmitted data is sampled and the sampled data is used to cancel the echo.

Echo cancellation used with the hybrid.

Echo cancellers are adaptive digital filters that estimate the echo for a given transmitted signal based on the knowledge of a transmission lines characteristics. The transmission line characteristics must first be known and this is ascertained before data transmission begins by monitoring a training sequence sent to the line. As data transmission takes place, the echo canceller adapts to maintain a replica of the echo. The echo is then subtracted from the received signal to reduce the error. The principle is shown in figure 3. Roessler has presented an ISDN echo-canceller implementation in [10].

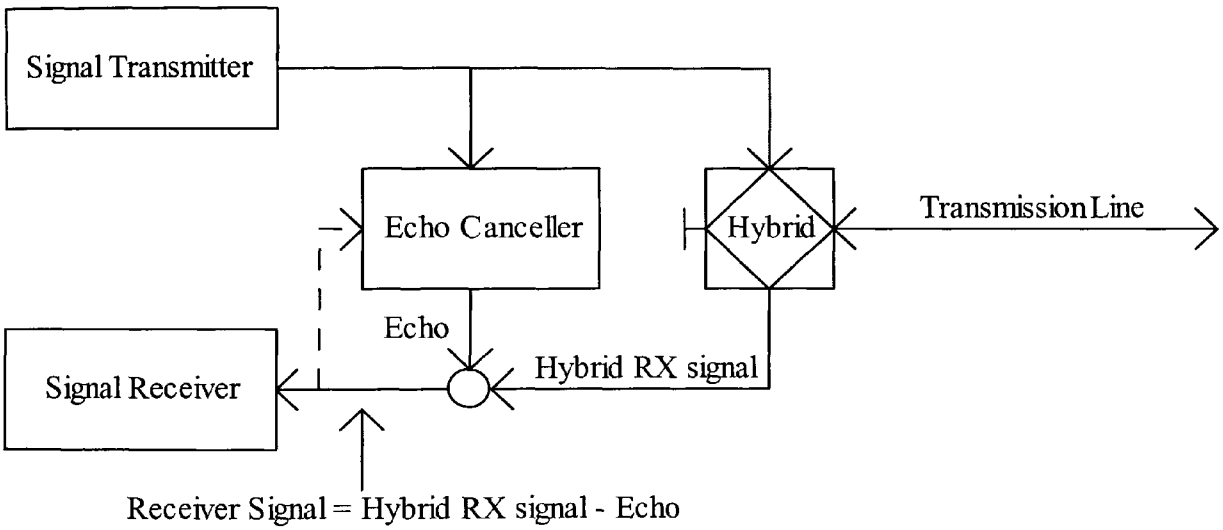


Figure 3 - The echo-cancellation principle with hybrid for ISDN.

Crosstalk.

Crosstalk is a form of signal interference that affects adjacent twisted pairs within a cabling system. The signal in one twisted pair may induce a signal in another pair via

electro-magnetic coupling. Two forms of crosstalk can be identified and represented graphically by figure 4.

The signal source is generating a signal on pair A, but because of its proximity to pair B, electro-magnetic coupling between the two pairs occurs. Some signal energy crosses from the signal source end of the cable to the hybrid in the near end receiver connected to pair B. This form of crosstalk is termed near-end-crosstalk (NEXT). The signal on pair A may also be coupled to the hybrid in the receiver at the far end of pair B and this form of crosstalk is termed far-end-crosstalk (FEXT). Crosstalk may interfere with the echo-cancellation process if the coupled signal has sufficient energy.

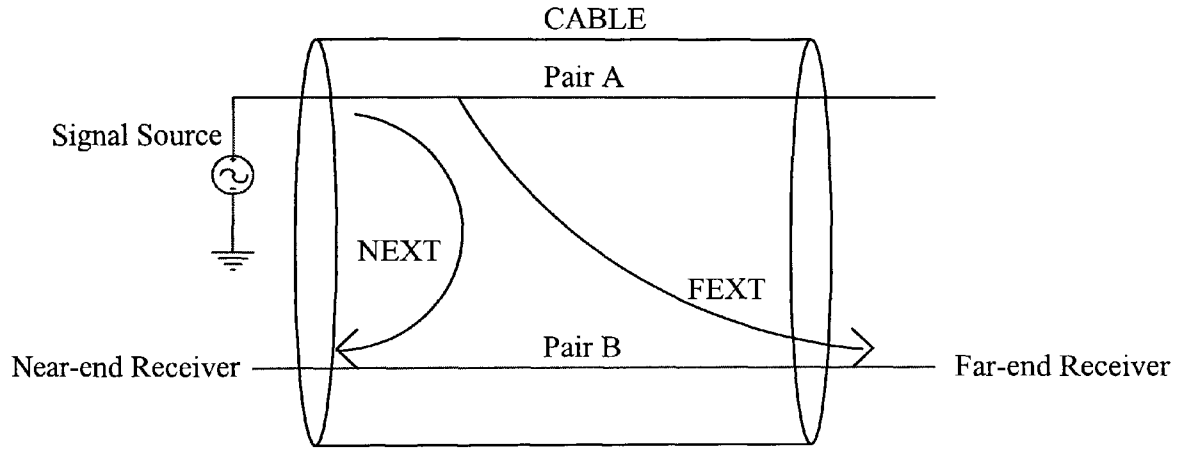


Figure 4 – NEXT and FEXT.

Methods to produce new models for Crosstalk in mixed DSL environments has been presented by Galli, Valenti and Kerpez [11], [12]. Different DSL technologies deployed on the same cable binder could potentially interfere with one another. Interference due to NEXT is particularly a problem in fully symmetrical transmission systems that use Hybrids with echo-cancellation as both transmit and receive spectra completely overlap. Galli, Valenti and Kerpez have published the

simulated results of NEXT from 49 BR-ISDN disturbers in an ISDN binder group [13]. It has been stated that the maximum data transmission distance that can be achieved in the presence of NEXT is 18000 feet for basic rate ISDN [4].

Line codes used for ISDN data transmission.

The line code adopted for basic rate ISDN in North America, Japan and the UK is called Two Binary One Quaternary (2B1Q). 2B1Q is a pulse amplitude modulated waveform and is presented in figure 5. 2B1Q coding uses one quaternary symbol, called a Quat, to represent two bits of binary data. Therefore, only half the bandwidth is required for a given data rate. The symbol levels of -3, -1, +1 and +3 correspond to RMS voltage levels of -2.5V, -0.83V, +0.83V and +2.5V. The 2B1Q waveform shows the symbol levels as -1, +3, +1, -3, +3, and -3. The data stream 011011001000 is used to modulate the waveform.

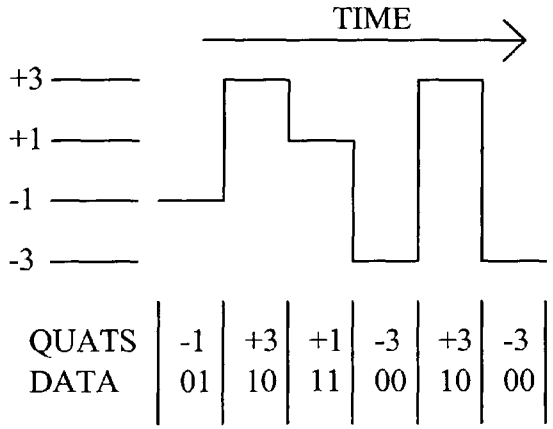


Figure 5 – A 2B1Q waveform.

The factors considered during the choice of the 2B1Q line code for Basic Rate ISDN are outlined in [14]. European countries such as Germany and Spain have adopted the Four Binary Three Ternary (4B3T) line code. Walters has described this

coding technique in [15]. Buchner has also presented an extensive description of Ternary line codes in [16].

Project number 1 will deal only with the 2B1Q line code. DSL Line codes can be characterised by their Power Spectrum Density (PSD). The PSD is defined in the T1E1.4 Spectrum Management Standard [6] as the power level and frequency content of a signal. Ifeachor and Jervis have presented a detailed description of Spectrum estimation and analysis in [17] and have defined the power spectrum density as,

$$PE(\omega) = \sum_{m=-\infty}^{m=\infty} C_{xx}(m) e^{-j\omega m} \qquad (1).$$

Where, $PE(\omega)$ is the Power Spectrum Density, $C_{xx}(m)$ is the autocovariance function of a discrete data sequence X(n) where n is the index of each sample.

The PSD is plotted with decibels against frequency in Hertz. Decibels in this case are referenced to one mili-watt and normalised to the resolution bandwith of the measurement in Hertz (dBm/Hz). Figure 6 presents an example of three PSD plots superimposed on the same graph. In each case the PSD of a 2B1Q signal operating at different signaling rates is plotted. The Basic rate ISDN service operates at a data rate of 144kbps ((64kbps x 2) + 16kbps). A further 16kbps is dedicated to maintenance and framing information bringing the total bit rate to 160kbps. The symbol rate for Basic Rate ISDN is 80Kbaud and the PSD shows power nulls at multiples of the symbol rate. In this case nulls appear at 80kHz, 160kHz, 240kHz and so on. In the practicable implementation designers filter the output of transmitters such that signal power above 160kHz is severely attenuated. The PSD for symmetrical digital

subscriber line (SDSL) services operating at 400kbps and HDSL services operating at 784kbps are also included on the plot.

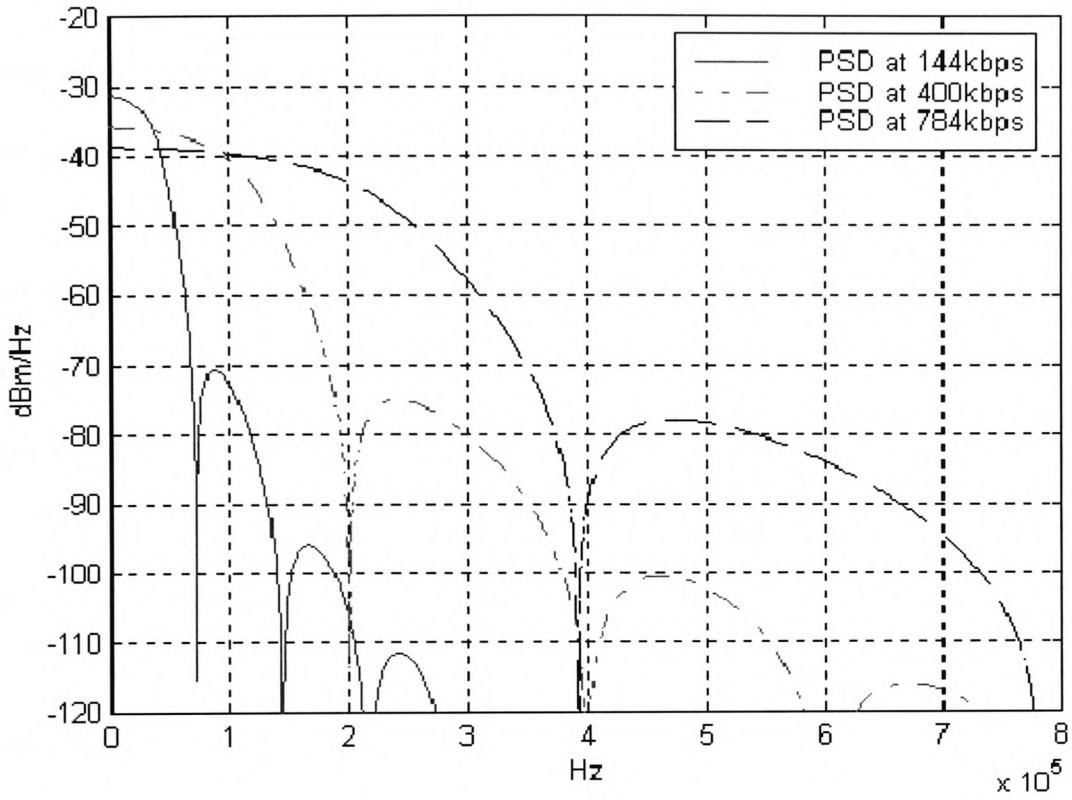


Figure 6 - PSD of 2B1Q line code at different signalling rates [7].

Non-parametric spectral estimation techniques.

Non-parametric spectral estimation methods include the Periodogram, the modified Periodogram, the Bartlett and Welsh modified Periodograms and the Blackman-Tukey method. Non-parametric methods have the advantage of achieving computationally efficient results using the Fast Fourier transform. The Discrete Fourier transform (DFT) of a finite duration of samples is defined as follows,

$$X(k) = \sum_{n=0}^{N-1} x(n).e^{-\frac{j2\pi nk}{N}} \qquad (2)$$

$X(k)$ is the Fourier transform, $x(n)$ is the discrete sampled signal where n is the index of each discrete sample, k is the harmonic number of the transform component and N is the number of consecutive samples of the signal.

The Fourier transform is symmetrical about the Nyquist frequency, which is half the sampling frequency. For example, lets say 512 samples are taken at a sampling frequency (f_s) of 512kHz, a Fourier transform is performed on the sampled data sequence and the amplitude spectrum is plotted against frequency. The spectrum will be symmetrical about 256kHz and the amplitude spectrum from 0Hz to 256kHz will be a replica of that from 256kHz to 512kHz.

The Fast Fourier transform (FFT) takes advantage of the computational redundancy in the Fourier transform to reduce the number of calculations required to determine the spectrum. A 512 point DFT would require 262144 complex multiplications and 261632 complex additions. Algorithms employed to achieve this reduction are outlined in [16] and it was stated that the DFT computations could be reduced to 2304 complex multiplications and 4608 complex additions when the FFT is used. The Xilinx 4010XL/Spartan FPGA used in the measurement hardware can only store 512 points of data at a resolution of 16 bits due to size constraints. The example above would provide a frequency resolution (Δf) or FFT frequency bin of 1kHz calculated as follows.

$$\Delta f = \frac{f_s}{N} = \frac{512000Hz}{512} = 1000Hz \qquad (3).$$

Where f_s is the sampling frequency and N is the number sampled data points.

The objective of the PSD measurement procedure is to display the PSD over a range of 0Hz to 240kHz while maintaining a frequency resolution of 1kHz. The frequency resolution is defined in ANSI T1.601 [4]. A sampling frequency of 512kHz allows a frequency range of 0Hz to 256kHz to be displayed. The FFT requires the number of samples in the data sequence to be a power of 2. The selection of 512kHz as the sampling frequency and 512 samples as the sample size satisfies all the measurement requirements. Parseval's theorem states that the average power in the time domain is the same as that in the frequency domain and is related by the following equation as derived in [18].

$$P_{ave} = \frac{1}{N}\sum_{n=0}^{N-1} x^2(n) = \frac{1}{N^2}\sum_{k=0}^{N-1}\left|X(k)\right|^2 \qquad (4).$$

N is the number of discrete samples in the time domain and in the frequency domain. $x(n)$ is a sampled time domain sequence. $|X(k)|^2$ is the power spectrum density and can be expressed as,

$$\left|X(k)\right|^2 = X(K).\widetilde{X}(K) \qquad (5).$$

$\widetilde{X}(K)$ is the complex –conjugate of $X(K)$. The power spectrum depends only on the magnitude spectrum and is independent of the phase spectrum. If $x(n)$ represents a voltage waveform then the units of the power spectrum density will be volts squared.

If it is desired to present the result in Watts then the resistance value across which $x(n)$ was measured must be included in the calculation. The application note [19] from National Instruments outlines the practical considerations in FFT implementation and power spectral analysis. Non-parametric methods used in spectrum analysis may suffer from spectrum distortion caused by inadequate sampling rate, picket-fence effect, spectral smearing or leakage and trends in the sampled data. The term Device Under Test (DUT) is used in reference to the equipment being tested. In project number one, a 2B1Q signal generator was developed to simulate a DUT. The resulting 2B1Q signal was measured to produce the PSD plot. The 2B1Q signal generator was configured to produce a random signal over the period that the measurement was taken so that problems due to waveforms trends were avoided.

Sampling Rate.

In this case the sampling rate is defined by the required resolution and the maximum number of FFT data points. In this case a sampling rate of 512kHz is more than twice the maximum frequency of the 2B1Q waveform to be measured. The frequency of the 2B1Q waveform is 80kHz and therefore the nyquist criterion has been satisfied.

Picket-fence effect.

The inability to differentiate between two frequency components in the measured spectrum is termed picket-fence effect or scalloping loss. Scalloping loss (SL) has been defined by Harris in [20] as "a representation of the maximum

reduction in processing gain, which occurs mid-way between the harmonically related frequencies". S.L is related to the resolution or frequency bins of the FFT used to calculate the spectrum. If the frequency difference between two frequency components is less than the frequency resolution then these two frequency components cannot be represented properly. The energy in these frequency components is shared between neighboring harmonics at the edges of the frequency bin. The two frequency components are decreased in amplitude and the amplitude of the harmonics at the edges of the frequency bin is increased. Increasing the number of data points for a given sampling frequency increases frequency resolution and helps to prevent this problem. Augmenting zeros or zero padding may be added to the sampled data sequence to achieve this. This serves to smooth the spectrum but does not add any additional information to the spectrum. In this case the resolution requirement for the spectrum measurement is 1kHz and is defined by the T1.601 standard and the T1E1.4 spectrum management standard.

Spectral Leakage.

When a data stream is sampled over a finite period of time the effect is to gate the data stream with a rectangular pulse. This can be regarded as multiplying the data stream by a rectangular window in the time domain or by convolution of the sampled data stream with a rectangular window in the frequency domain. The spectrum of a rectangular window consists of a main frequency lobe and an infinite number of side lobes. The effect of the side lobes from the rectangular windows spectrum is to introduce spurious peaks in the convolved frequency spectrum, which causes distortion. These spurious peaks may contribute unwanted power to the calculated

power density spectrum thus reporting erroneous power levels. Employing a window with a frequency spectrum that has side lobes with low amplitude levels can minimize distortion [17].

Windowing.

If the spectrum of a sampled data sequence containing 512 points is to be calculated a window containing 512 values is required. Each point in the sampled data sequence is multiplied by its corresponding window value. If the 512 values of a window are denoted N=0 to N-1 then at N/2 the value should be 1 and the window values should taper off to zero on either side towards N=0 and N-1. This has the effect of attenuating the sampled data values at both ends of the sequence and hence reduces the signal power. In this project windowing is performed in the time domain, but may also be performed in the frequency domain. Harris has defined different windows and has stated that the appropriate choice of window is dependant on properties such as equivalent noise bandwidth (ENBW), processing gain, processing loss, and minimum resolution bandwidth [19].

The shape of the window determines the equivalent noise bandwidth. If the window is regarded as a spectral filter, then ENBW is defined by Harris as the bandwidth of an ideal rectangular filter, which passes the same amount of applied white noise as the spectral filter in question [19]. The processing gain or coherent gain is defined by Harris as the ratio of output signal-to-noise ratio subsequent to windowing to the input signal-to-noise ratio prior to windowing. The attenuation caused by the window at its edges represents a processing loss (P.L) and causes a reduction in the measured signal power. Harris has defined the ENBW and coherent

gain values for the different windows used in this project. Harris stated that the important properties in selecting a window are highest side lobe level and worst case processing loss. It was concluded that the Blackman-Harris and the Kaiser –Bessel windows are the best choice. The Kaiser-Bessel offers the ability to vary the main lobe width to side lobe level compromise by adjusting the numerical parameter α. Harris stated that an element of trial and error is involved in the determining of the value of α and this will be considered during this project. Harris also concluded that the Hamming, Hanning, cosine-tapered (Tukey) and Bartlett (Triangle) windows all have inferior performance. In this project different windows are used in the calculation of the spectrum and a comparison of the results will be performed.

Calculation of the PSD using non-parametric methods.

The classic method used to estimate the PSD from a sampled data sequence of N samples is the periodogram introduced by Sir Arthur Schuster in 1898. The periodogram is defined by Saeed V. Vaseghi in [21] as follows,

$$P_{xx}(f) = \frac{1}{N}\sum_{N=0}^{N-1} \left| X(f) \right|^2 \qquad (6).$$

It can be seen that this has already been derived from Parseval's theorem. Proakis and Manolakis [22] have also expressed the Periodogram as follows.

$$E[P(f)] = \sum_{m=-\infty}^{m=\infty} w_B(m) c_{xx}(m) \exp(-j2\pi fm) \qquad (7).$$

The term $c_{xx}(m)$ is the autocovariance of a sampled data sequence evaluated at lag m. f is the frequency. $w_B(m)$ is the triangular (Bartlett) window. These two definitions are consistent with the theory that the PSD can be derived directly from the sampled data sequence or from the Fourier transform of the autocovariance function. The sampled data sequence is a finite representation of what is a continuous signal. It is expected then that the calculated spectra of different sampled data sequences of the same continuous signal should vary about the average spectra. It was shown by Ifeachor and Jervis that the spectrum calculated by the Periodogram is not consistent from one realisation to the next and this variance does not decrease as the number of sampled data points is increased [17].

A single Periodogram will produce a rough estimate of the spectrum. Averaging a number of Periodograms has the effect of smoothing the estimated spectrum. Employing the Bartlett [23] or the Welsh method [24] can reduce the variance in the estimate of the PSD. The Bartlett method consists of segmenting a large number of sampled data points into sections. The Periodogram is calculated for each section and the calculated spectra are averaged to produce the final result. The Welsh method consists of segmenting a large number of sampled data points into overlapping sections and an average of the calculated spectra is produced. It was shown by Ifeachor and Jervis that as the length of the sampled data sequence is increased the variance tends to zero [17]. The Blackman-Tukey method is performed by calculating the autocorrelation of the sampled data sequence, applying a suitable window and then calculating the FFT directly. Proakis and Manolakis have compared the methods using a quality factor based on the ratio of the square of the mean of the PSD to its variance [22]. Their findings suggested that the Blackman-Tukey method offers the best quality factor. The Welsh method is the second best method. Practical

implementation of non-parametric methods requires a large amount of memory to store the captured data sequence. In this project, the Xilinx FPGA that was used has storage facilities for 512 samples at a resolution of 16 bits and provision of more memory would mean employing an FPGA with more internal SRAM or adding an external SRAM.. The implementation will concentrate on taking numerous samples and averaging the spectra using the modified Periodogram method.

2.2. Literature Review for Project Number 2.

In project number two both NT and TE circuit blocks were designed and implemented. The main document related to the project is Recommendation, I.430 [25], produced by the ITU-T. Recommendation I.430 defines the electrical interface and signalling for the ISDN S Interface point. The document is divided into sections dealing with each aspect of the S Interface. Project number two focused on two areas of Basic Rate ISDN design. Firstly, the design of a Basic rate ISDN TE transceiver with integrated Loopback control circuitry and PRBS circuits was completed. Secondly, a NT design was developed and implemented with the necessary features to facilitate verification and validation of the TE Transceiver. The Digital electronics for both the NT and TE design was implemented on an FPGA and the analogue electronics was implemented with commercially available active electronics. The Terminal Equipment design can be partitioned into three defined functional blocks. These are the S Bus Synchronisation and DPLL circuit, the Loopback controller with integrated PRBS circuit and the ISDN signal generator circuit.

The S Bus Synchronisation and DPLL circuit.

The Line code used on the S Bus is Pseudo-Ternary coding. The coding is performed by transmitting a binary one as a null voltage level and a binary zero as either a positive or negative pulse. Consecutive binary zeros are coded such that the polarity of the transmitted pulse is alternated with respect to the last pulse transmitted. This coding rule must be observed in all cases except for the generation of timing events. In an ISDN network the NT derives its timing from the exchange in the central office and the TE derives its timing from the signal transmitted by the NT. The timing events are embedded within the data frames from the NT. Two timing events are incorporated within each data frame to facilitate the synchronisation process. Recommendation I.430 defines a timing event as the transmission of two consecutive pulses with the same polarity. This is called a coding rule violation. The synchronisation procedure is referred to as frame alignment and is defined in section 6.3 of I.430. Throughout Project two, the term synchronisation is used instead of frame alignment. Synchronisation is complete when a TE has identified three consecutive sets of coding rule violations obeying the frame alignment rules.

The data on the S Bus will be valid at the centre of each bit within a frame. The FPGA master clock is therefore required to sample the S Bus data at this point in time. The sampling clock is derived directly from the FPGAs external 15.36MHz crystal. The resolution of the clocks are defined by I.430 as 100 parts per million for both TE and NT. During the synchronisation process the TE relies on the resolution of the clock to ensure that data is sampled at the correct time. Once the synchronisation circuit has established synchronisation with the S Bus the sampling clock will be roughly coincident with the centre point of each data bit on the S Bus. A Digital Phase

Locked Loop (DPLL) is then enabled to perform adjustment of the sampling clock edge so that this relationship is maintained. The only consistent timing event in an S bus data frame is the relationship between the FL pair. The position of the FL pair can only be determined when synchronisation is established. The F and L bit are defined in section 5.4 of I.430. The L bit is always the opposite polarity to the F bit. The consistent change in polarity between the F and L bit generates a reference edge for the DPLL. The DPLL compares the FL pair reference edge against the locally generated sampling clock once every frame and makes an adjustment to the sampling clock when required. The synchronisation circuitry and the DPLL circuitry are both features of the FPGA design.

There has been no significant academic work completed that directly relates to the design of DPLL circuits for Basic Rate ISDN applications. The majority of work this has been completed by Industry and thus the design methods remain the property of commercial organisations and do not appear in the public domain. The work to date has concentrated on implementation of DPLL solutions with mixed signal ASSPs for Primary rate applications. A fully integrated DPLL for a Primary Rate application is presented by Harufusa et al [26]. The design of the DPLL circuitry for project number two is integrated with the S Bus synchronisation circuitry. The S Bus synchronisation circuitry monitors the FL pair and derives the reference edge for the DPLL. The DPLL in this project is composed solely of digital blocks. The DPLL can be partitioned into two components, the Phase Detector (PD) and the Digitally Controlled Oscillator (DCO).

The DCO generates the master clock for the design and is derived from a crystal oscillator connected to the FPGA. The DCO produces a fixed frequency of 192kHz and this is used as the clock signal for the rest of the FPGA circuitry. The

objective of the DCO control circuitry is to move the DCO clock edge and hence the clock phase relative to the FL pair reference signal produced by the synchronisation circuitry. An extensive treatment of DPLL design techniques has been presented by Best [27]. Best outlines the different types of Phase detectors and DCOs. In project number two the principles outlined by Best are followed and a phase/frequency (Type 4) detector and an Increment-decrement (ID) VCO design was developed to produce a fixed frequency and variable phase DPLL.

The Loopback controller and integrated PRBS circuits.

The NT provides loopback facilities for the B1 and the B2 channel. A TE may request a loopback by transmitting a loopback request signal on the maintenance channel toward the NT. The NT monitors the maintenance channel and responds to a loopback request by retransmitting the data from the TE back to the TE in the next frame. Once a loopback is active the NT transmits a loopback indication signal on the maintenance channel towards the TE. The loopback indication signal is transmitted while the loopback remains active. A loopback may be placed on B1, B2 or B1 and B2 simultaneously. The data that flows from the NT towards the TE is termed the NT-to-TE direction of transmission. The data that flows from the TE towards the NT is termed the TE-to-NT direction of transmission. The maintenance channel for the NT-to-TE direction is called the S-channel and the maintenance channel for the TE-to-NT direction is called the Q-channel. Section 6.3.3 of Recommendation I.430 describes both maintenance channels in detail.

The Layer 1 maintenance facilities between an NT and TE are presented in section 7 of Recommendation I.430. The methods described therein to establish a

loopback at the NT have been followed during the design exercise. Once a Loopback is established the TE begins transmitting a PRBS pattern in the NT-to-TE direction. The NT loops the pattern back to the TE, which then monitors the number of errors in the received data stream. The TE maintains a count of the number of bit errors. The BER is derived from the periodic evaluation of this count value. The BER requirements and methods employed to calculate BER for Basic rate ISDN are detailed in the ITU-T Recommendation G.821 [28]. The design and implementation objectives do not include the development of the microcontroller code to evaluate the BER. The work concentrated on integrating the PRBS generator and Receiver circuitry with the maintenance channel control circuitry to reduce to complexity of current implementations.

The design of PRBS generator circuits is based on Linear Feedback Shift-Register (LFSR) theory. LFSR theory has been well presented by B.R Wilkins [29]. A Pseudo-Random Binary Sequence is generated by an N-stage shift-register. A combination of the N-stage shift-register outputs is connected to an exclusive-or gate. The exclusive-or gate output signal is fed back to the input of the shift-register to complete the circuit. The number of shift-register stages defines the pattern length. For example an eleven-stage shift-register produces a random pattern length of 2047 (2^{11} –1) before the sequence repeats. Section 5.2 of the ITU-T Recommendation O.150 [30] specifies the implementation details for PRBS circuits utilised for BERT on channels supporting bit rates of 64kbps. Recommendation O.150 states "The sequence may be generated in an eleven-state shift register whose 9^{th} and 11^{th} stage outputs are added in a modulo-two addition stage, and the result is fed back to the input of the first stage". This implementation method was employed to produce the PRBS circuits for Project Number 2. The PRBS receiver circuit is based on the same

circuit as that used for the PRBS generator. The design is more complex to design and implement and has been explained fully in Report number 2.

The ISDN Signal Generator.

The ISDN generator is composed of analogue and digital components. A signal Generator is required for both the TE and NT designs. The digital component for the TE design is implemented on an FPGA and performs the S Bus data framing and signal coding for the B, D and Q channels. The Signal Generator for the NT design is implemented in the same way and performs the same tasks. In the case of the NT design, the provision for the S channel is made. Section 5 of Recommendation I.430 defines the frame structure and coding rules for both NT and TE. Section 6.3.3.2 of I.430 defines the position of the Q channel within the transmitted data frame towards the NT. In the same way, section 6.3.4 defines the position of the S channel. The analogue component performs the differential signal generation for application to the S Interface point. Section 8 of I.430 specifies the electrical characteristics for Basic Rate ISDN equipment.

2.3. Literature Review for Project Number 3.

Project number 3 focused on the design and development of circuitry required for ISDN protocol monitoring on the S Interface. The protocol monitor provides a technician with detailed information of the signal activity at each layer in a Basic Rate ISDN network. The signals that are exchanged between an NT and TE during the layer 1 activation stage are defined in section 6 of Recommendation I.430. The

monitor was designed to identify the INFO0, INFO1 and INFO3 signals in the TE-to-NT direction and the INFO0, INFO2 and INFO4 signals in the NT-to-TE direction. In Section 6 of I.430 a state matrix is used to describe the activation and deactivation procedure for Layer 1. The monitor was also designed to extract the D channel data from the data frames flowing in both directions of data transmission for onward transmission to a Microcontroller where the decoding would take place. The Microcontroller code was not part of the development work for Project number 3.

The D channel is used to transmit call activation messages between an NT and TE. Layer 2 and Layer 3 messages are encoded within the D channel. The protocol employed on the D channel is the Link Access Protocol for the D Channel (LAPD) and is based on the Higher-Level Data Link Control protocol. The operation of LAPD has been described by Shatz and Kajka [31]. A more detailed description of the operation of LAPD including the frame structure and the format of fields is presented in the ITU-T Recommendation Q.921 [32].

Project number 3 employed the same S Bus Synchronisation and DPLL circuit that was already developed in Project number 2. In Project number 3, the S Bus Synchronisation and DPLL circuit was used to synchronise to both the NT-to-TE and TE-to-NT directions of data flow simultaneously. Once synchronisation with the network is established the D channel data from both directions of data flow can be extracted and decoded to obtain the Layer 2 and Layer 3 messages. The design and implementation of the circuitry required to identify the Layer 1 signals, extract the D channel data, buffer the data and transmit it to a HDLC decoder comprised the main bulk of the work. A HDLC decoder is required to decode the D channel data into Layer 2 and Layer 3 messages. Industry has produced many HDLC controllers ranging from Microcontroller integrated systems to FPGA and Complex

Programmable Logic Device (CPLD) based solutions. A flexible CPLD based HDLC controller implementation has been proposed by Leigh, Fingeroff and Morse [33]. The Motorola 68302 was chosen, as it is equipped specifically for ISDN and would be a realistic choice for Basic Rate ISDN testers. The Motorola 68302 microcontroller can perform two HDLC decoding operations simultaneously. The 68302 microcontroller has been widely adopted by designers producing Basic Rate ISDN products and a data sheet can be obtained from Motorola's web site. The 68302 microcontroller uses the ISDN Orientated Modular Interface – Revision two (IOM-2) to communicate with other ISDN devices. The IOM-2 Interface was originally conceived by Siemens and has been widely adopted by Industry to date An IOM-2 Interface was incorporated within the design and was used to transmit the D channel information to the 68302 microcontroller. The IOM-2 Interface is defined by the Siemens IOM-2 Manual in [7] and was used as the reference for the IOM-2 Interface design task.

3. Achievements.

The reconfigurable hardware platform that was designed and implemented successfully supported the three portfolio projects. The development process succeeded in incorporating the required hardware facilities for each project within a single hardware platform. A block diagram for the hardware platform is shown in figure 7. The hardware platform utilises an FPGA as its core component. The FPGA is reconfigured to define the system operation. The FPGA was chosen as it offers speed advantages over a flash based microcontroller implementation. The hardware platform offers an ISDN equipment designer a blank sheet on which to develop digital design solutions for Basic Rate ISDN applications. The platform is limited to data features, however, the addition of a Codec would extend the facilities to voice services. The hardware platform may be connected to the S Interface point to support ISDN tests on the subscribers side or the U Interface point to support tests across the ISDN network. The FPGA may be reconfigured any number of times to provide different system functionality.

Three projects were chosen to demonstrate to effectiveness of this design technique over design methods currently adopted by industry. Each project dealt with the development of one ISDN test feature commonly used by ISDN technicians. Engineers use the PSD measurement feature in assessing the output power of ISDN transmitters. PSD measurement features are also commonly employed in crosstalk interference identification equipment. The BERT feature is an essential test facility offered by Basic Rate ISDN testers used to install, commission and test ISDN networks. The ISDN protocol analyser is commonly used by technicians to identify complex faults on ISDN networks.

The PSD measurement feature utilises the data acquisition circuit shown in figure 7 to digitise the 2B1Q waveform at the U Interface. The BERT feature successfully utilised the TX analogue circuit to transmit the framed and coded ISDN signals. The signals were transformer coupled to the S Interface point and the S Bus RX analogue circuit successfully extracted the analogue signals from the S Interface point for application to the FPGA. When the hardware platform is configured as an ISDN Protocol Monitor the extra S Bus RX analogue circuits allows the FPGA to extract the signals from both directions of data transmission simultaneously. The analogue electronics to achieve this was successfully tested and validated. The digital electronics was successfully simulated using a VHDL test bench.

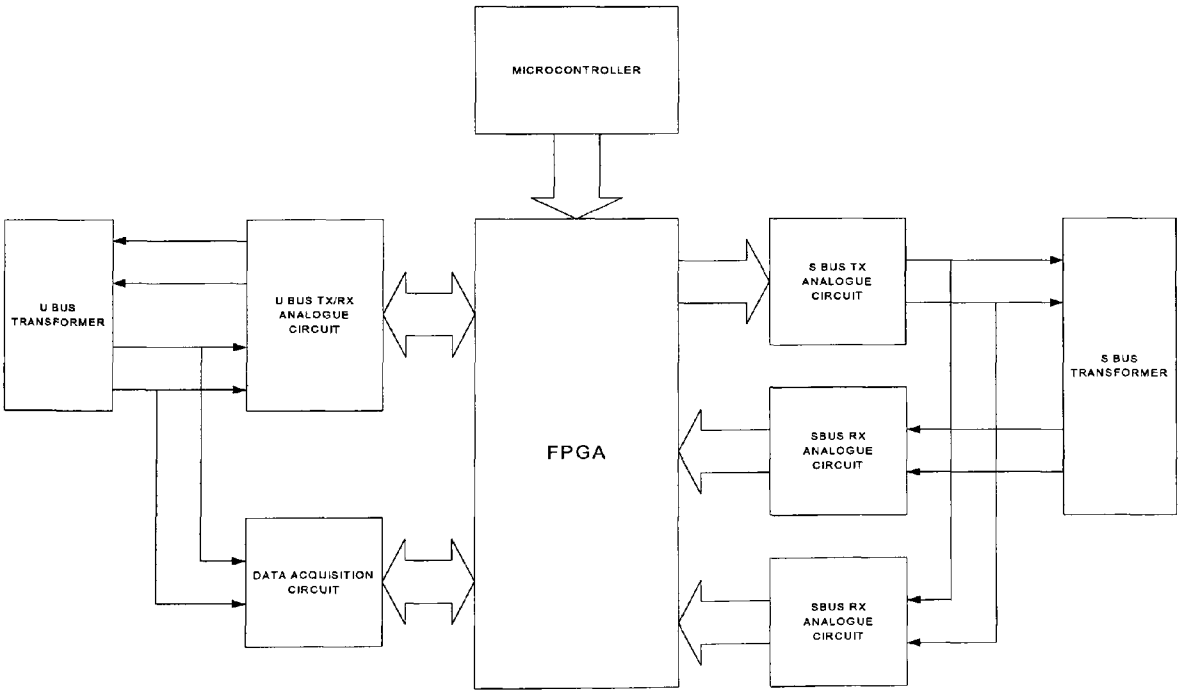


Figure 7 - The Reconfigurable Hardware system block diagram.

The three test features were successfully realised on the hardware platform as three different system configurations. Test equipment currently available on the market does not provide the same flexibility demonstrated here. It is common to find that separate testers are required to provide the same functionality possible with the hardware platform developed for this portfolio. This design approach achieved the successful integration of the ISDN test features within one hardware platform. Figure 8 shows the actual PCB with the FPGAs fitted. Two PCBs were developed to separate the digital electronics from the sensitive analogue electronics. Figure 9 shows the analogue PCB.

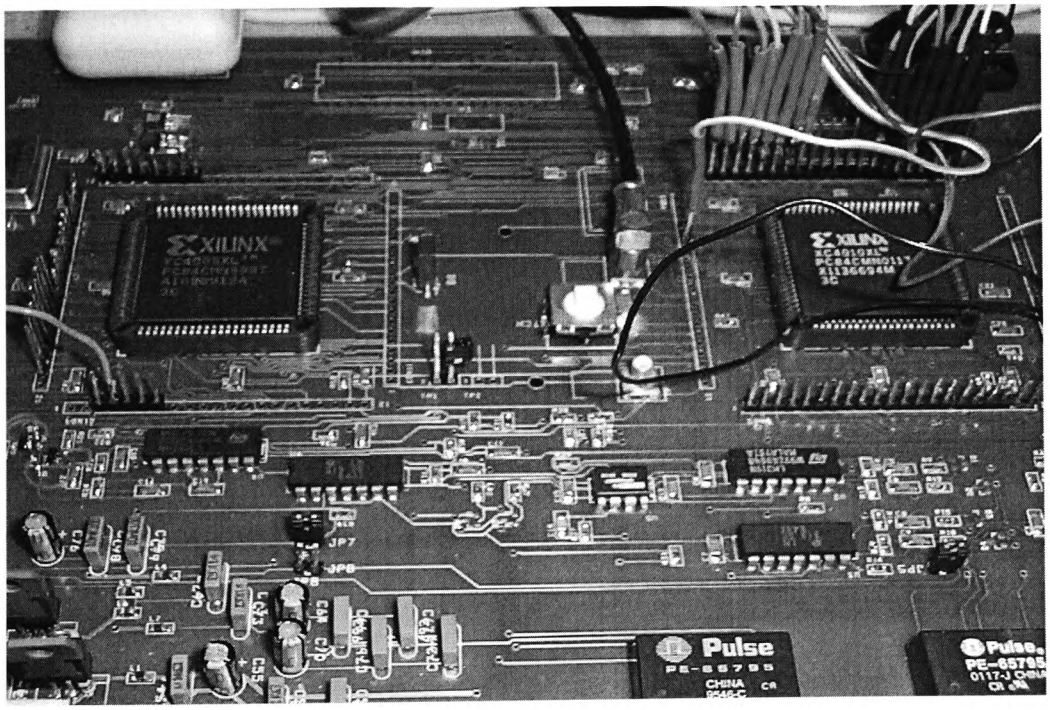


Figure 8 – The FPGA PCB.

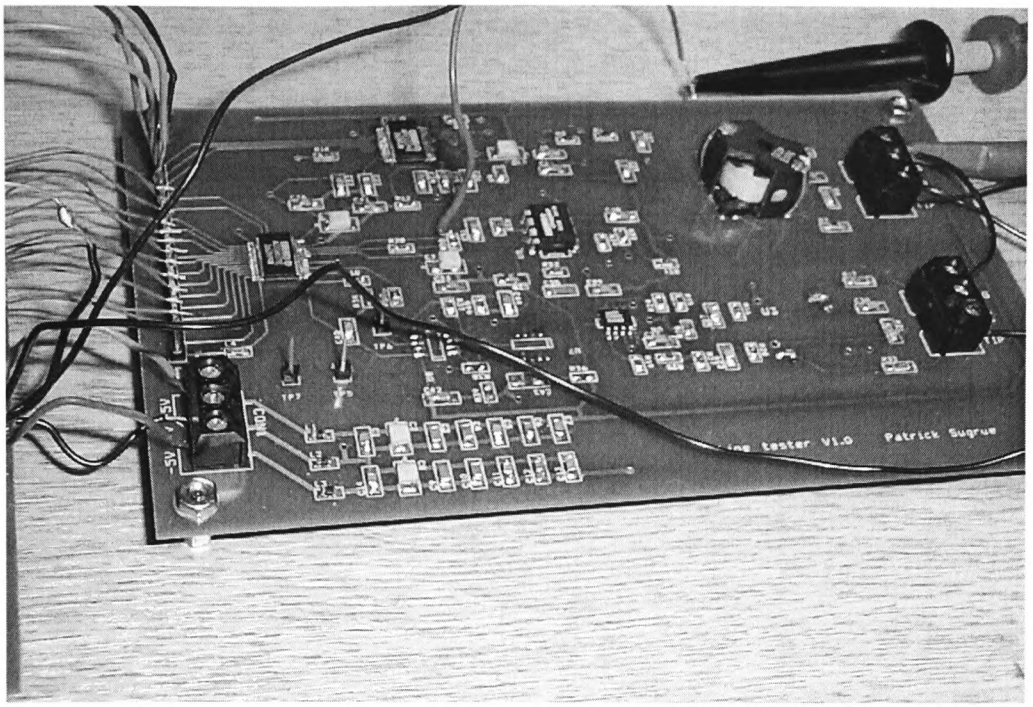


Figure 9 - The Analogue PCB.

Facilities for two FPGAs were made on the PCB. In Project number two, one FPGA is configured as a TE and functions as a BERT tester. The second FPGA is configured as the NT. The TE communicates with the NT in order to establish a loopback. The NT was designed to provide the maintenance channel operation and loopback facilities required to test and validate the BERT design. The block diagram for the NT design is presented in figure 10. The NT is again transformer coupled to the S Interface point in the same way as the TE in order to provide a realistic test environment.

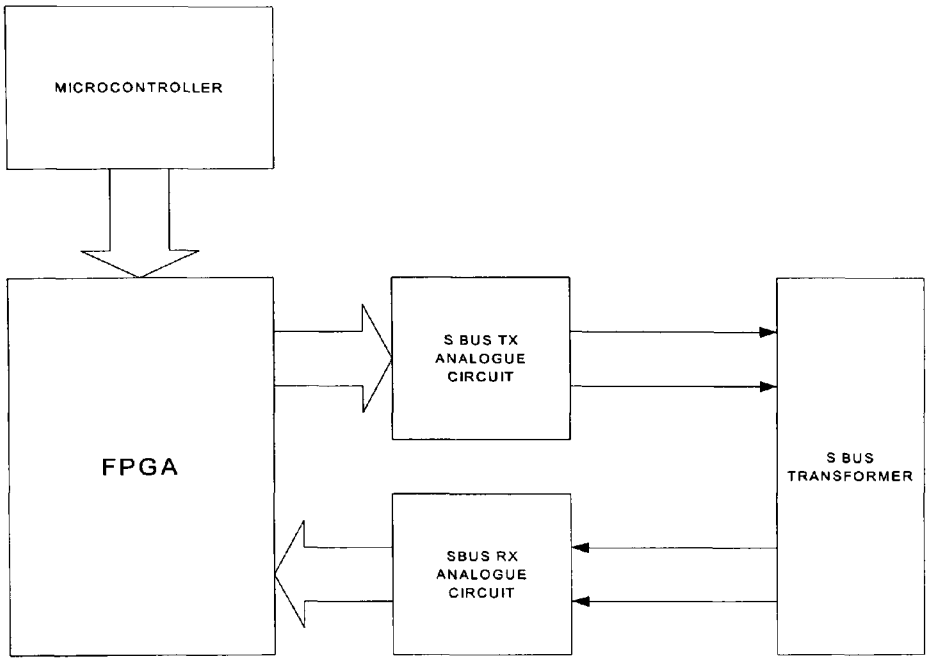


Figure 10 - The NT hardware block diagram.

FPGA Design Flow.

The Portfolio designs were targeted at Xilinx FPGAs. The designs were simulated with the ModelSim digital simulator produced by Model Technologies. FPGA design implementation was achieved using the Xilinx Flow Engine (version M1.5.19) software package. The design process utilised a mixed schematic entry and VHDL implementation approach. A schematic diagram representing the top-level block diagram and associated hierarchy was produced using the schematic capture facility offered by Foundation (version F1.5). Each block in the hierarchy has a VHDL implementation associated with it. The VHDL synthesis was performed by VHDL-Express (Version 3.1.1.0w).

The FPGA design flow is shown in figure 11. Once the VHDL code is written VHDL compilation is performed before digital simulation can begin. The designer may alternate between the design stage and the functional simulation stage and iterate

toward a working design. When the design is complete, a suitable FPGA is targeted for implementation and the design is synthesised. During Synthesis the VHDL hardware description is translated into a digital circuit to reflect the functionality defined by the designer. Once Synthesis is successfully completed the FPGA implementation flow is invoked.

The FPGA implementation flow is shown in figure 12. The process is broken down into five stages. The Translate process performs logic reduction and optimisation. The mapping process maps the optimised logic into the FPGAs configurable logic blocks (CLB). The place and route process connects the CLBs together to realise the circuit. The timing process compares the actual timing due to the logic and routing delays against a timing file defined by the designer, and provides a timing report. If the required timing is not achieved the designer will be required to return to the implementation stage or choose a faster FPGA. The last process in the flow is to create a configuration bit stream file. This file is then used to configure the FPGA.

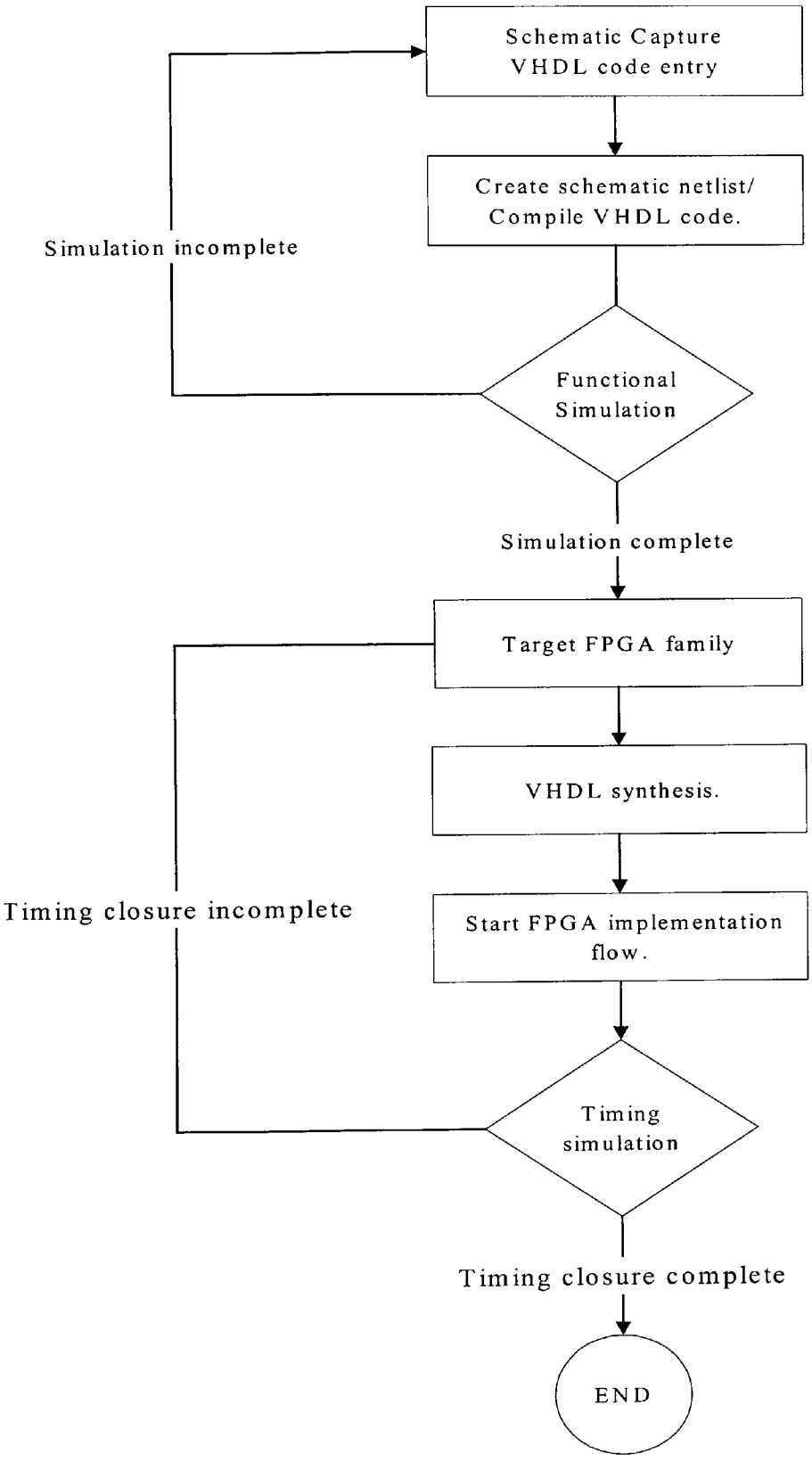


Figure 11 – FPGA Design Flow.

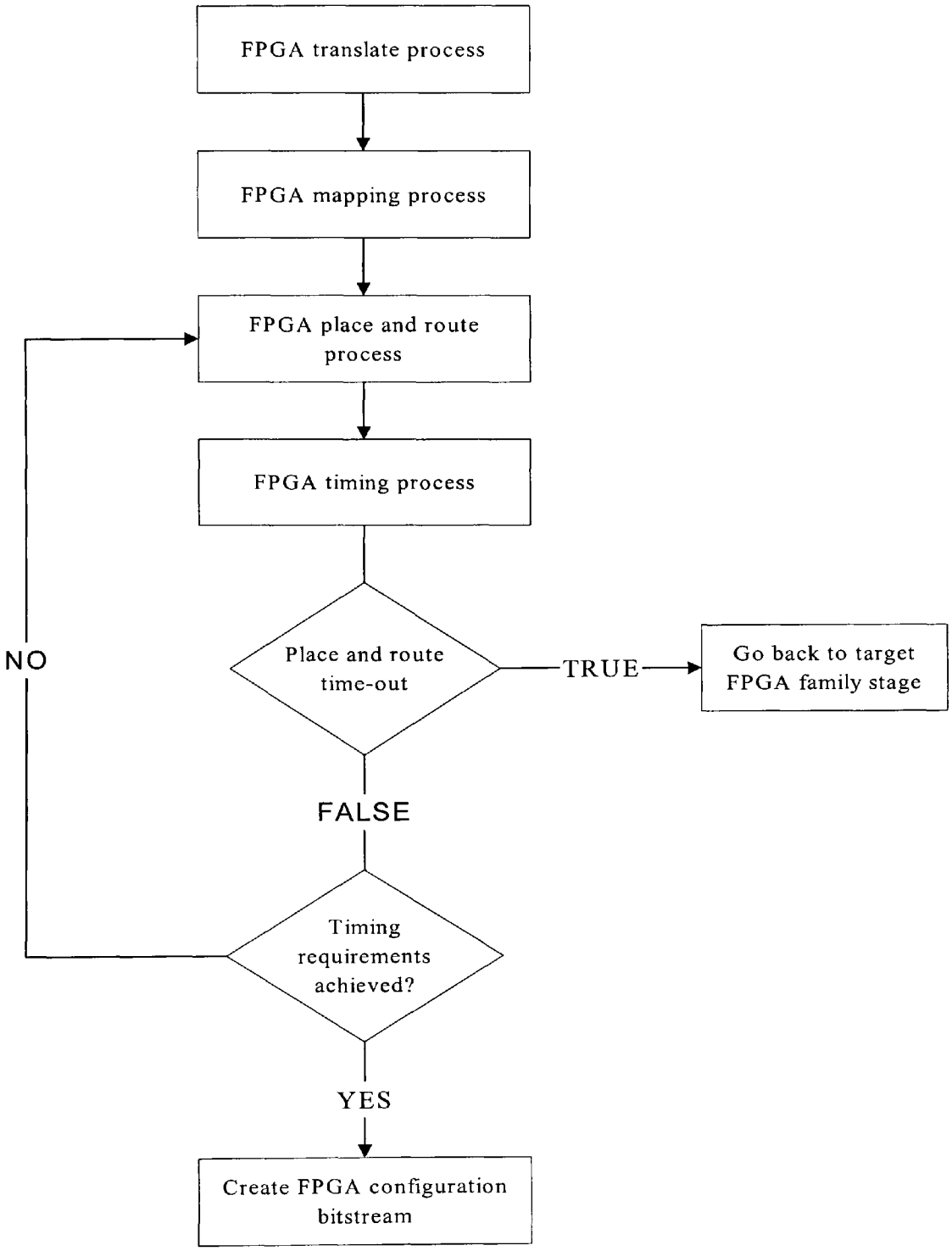


Figure 12 – FPGA Implementation Flow.

4. Conclusions and Further work.

The main objective of the portfolio was to demonstrate that FPGA technology could be utilised to reduce cost and increase integration and flexibility for Basic Rate ISDN test equipment designs. The development of a reconfigurable hardware platform was successfully achieved and it was demonstrated that FPGA technology can be employed to effectively integrate some of the essential ISDN test features within a single hardware platform. Three ISDN test features were developed as individual projects. The three projects were successfully completed and the objectives associated with each project were achieved. The design approach demonstrated a novel concept offering distinct advantages over the design methods currently employed by Industry. ISDN systems designers have not been successful in the implementation of an equivalent system that offers the same integration possibilities with the same flexibility as that presented in this portfolio and therefore, the techniques employed to achieve this constitute a significant contribution to knowledge in the field. From the work presented in this portfolio the following conclusions can be made.

4.1. Conclusions.

1. The reliance on third party vendors to supply essential ISDN chip sets is avoided with the FPGA design approach. The ownership of design components remains the property of the ISDN equipment designers and therefore, design changes demanded by the market can be easily achieved. The ISDN equipment designer may build a library of reusable digital components

that would over time contribute to reduced development time for new products. The ability to reconfigure the FPGA design and iterate towards a complete working design reduces the risk associated with ASSP design and provides a level of flexibility not possible with ASSP implementations.

2. It is possible to integrate different test features within the same hardware platform. It was shown how a Xilinx FPGA was configured to provide a PSD measurement, a BERT, and ISDN protocol monitor features. The reconfigurable nature of the hardware platform could be exploited to expand the test feature list to include other ISDN test facilities. The PCB design was developed to populate a Xilinx 4K/Spartan series FPGA. Atmel produce an FPGA with an equivalent footprint to that produced by Xilinx. The ability to populate the PCB with interchangeable FPGAs reduces the dependence on a single FPGA manufacturer.
3. The prime objective of project number 1 was to design and develop a cost effective PSD measurement feature for Basic Rate ISDN that can be easily expanded to facilitate the measurement of PSD above the Basic Rate frequency spectrum. It was demonstrated that a PSD measurement facility could be designed in a modular way so as to allow an increase in measurement resolution and measurement frequency. The PSD measurement feature was designed and implemented. Extensive tests were performed and it was concluded that the project objectives were achieved.
4. It was concluded in project number 2 that the design and development of an FPGA based S Interface transceiver with supporting analogue electronics was possible to achieve for Basic Rate ISDN applications. It was also concluded that the integration of the PRBS test features required for BERT applications

was successfully completed thus achieving the project objectives. The work successfully presented the FPGA design approach as an alternative to employing ISDN and PRBS ASSPs. It was shown how the cost of a BERT design feature was reduced and how the dependence on ISDN chip sets could be avoided by the adoption of FPGA technology.

5. The prime objective of Project number 3 was to show that the dependence on ASSPs to realise ISDN protocol monitors could be avoided with the adoption of FPGA technology. The design of a D channel extraction method and a data transportation method between the FPGA and an industry standard microcontroller was achieved. It was concluded in project number 3 that an FPGA based ISDN Protocol Monitor was a more effective implementation than the utilisation of two ISDN S Interface transceiver ASSPs thus achieving the project objective.
6. Project number 3 built on the success of project number 2. The S bus synchronisation circuits and PLL circuits were utilised again in project number 3. As logic designs are completed, a library of logic components can be built up over time. As the library grows, the design functionality is increased thus allowing designers to respond quickly with new designs as the markets demands. The design ownership remains firmly with the test equipment designer. This is not the case when the design is realised with ASSPs.

4.2. Further Work.

The reconfigurable hardware platform developed for this portfolio provides wide scope for future work. The design approach demonstrated within this work may be extended to provide extra test features for Basic Rate ISDN testing. The following tasks have been identified.

1 The FPGA design work can be extended to integrate the FFT and PSD measurement functionality. The FPGA may be upgraded to a device offering greater facilities. More RAM facilities can be employed to increase the number of samples that can be acquired. The PSD measurement variance associated with the periodogram can be reduced by adopting the Welsh method instead.

2 The work relating to the S Interface transceiver in project number 2 may be expanded upon to provide a Layer 1 activation and deactivation state machine, a D channel access controller and a B channel elastic buffering facility to provide a full-featured S Bus transceiver. The hardware platform also provides the hardware features required to facilitate a U Interface transceiver development project.

3 Project number 3 succeeded in the design and simulation of the layer 1 circuitry required for Basic rate ISDN protocol monitor realisation. The work can be continued to produce a physical implementation. The design was successfully placed and routed on a Xilinx 4010XL/Spartan FPGA. A Motorola 68302 microcontroller was identified as a suitable choice for implementation of the layer 2 D channel analysis. The HDLC functionality

may also be integrated with the FPGA circuitry, thus allowing the adoption of a cheaper microcontroller.

References.

[1] David E. Taylor, Jonathan S. Turner, John W Lockwood, Edson L. Horta, Dynamic hardware plugins: exploiting reconfigurable hardware for high-performance programmable routers, Computer Networks and ISDN Systems, vol. 38, no. 3, pp 295-310, Feb 2002.

[2] Stefanopoulos, N.V, Tombros, S.L, Liveris A.D and Stassinopoulos G.I, A generic ISDN terminal architecture for BRI and PRI applications, Proceedings from the Third IEEE Symposium on Computers and Communications, pp. 390 – 394, 1998, ISBN: 0-8186-8538-7.

[3] J.Dunlop and D.G. Smith, Telecommunications Engineering, Stanley Thornes (Publishers) Ltd, ISBN 0-7487-4044-9.

[4] American National Standards Institute, Integrated Services Digital Network (ISDN) – Basic Access Intreface for use on Metallic Loops for application on the Nework Side of the NT (Layer 1 specification), T1.601-1992.

[5] Stefano Galli, David L. Waring, Loop Makeup Identification via Single Ended Testing: Beyond Mere Loop Qualification. IEEE Journal on Selected Areas in Communications, vol. 20, No. 5, pp 923 to 935, June 2002.

[6] Coiffi, Silverman, Starr, Digital Subscriber Lines, Computer Networks Journal, vol 31, No 4, pp 283-311, 25 Feb 1999.

[7] Draft American Standard under review by the American National Standards Institute, Spectrum Management For Loop transmission Systems, T1E1.4/2000-002R2

[8] Siemens, The IOM-2 Interface Reference Guide, Version 01.90, March 1991.

[9] Infineon Technologies. Data Sheet for the PEB 24902/ PEF 24902. 23-Jan-2001.

[10] Roessler B, Wolter E. CMOS Analog Front End of a Transceiver with Digital Echo Cancellation for ISDN. IEEE Journal of Solid-State Circuits, Vol. 23, No. 2, pp. 311-317. April 1988.

[11] Galli, Valenti, Kerpez, Methods of Summing Crosstalk From Mixed Sources-Part 1: theoretical Analysis. IEEE Transactions on Communications, Vol. 50, No. 3, pp. 453-461, March 2002.

[12] Galli, Valenti, Kerpez, Methods of Summing Crosstalk From Mixed Sources-Part 11: Performance Results. IEEE Transactions on Communications, Vol. 50, No. 4, pp. 600-607, April 2002.

[13] Galli, Valenti, Kerpez, "A Frequency-Domain Approach to Crosstalk Identification in xDSL Systems", IEEE Journal on selected areas in Communications, Vol. 19, No, 8, pp. 1497-1506, August 2001.

[14] Lechleider W, "Line Codes for Digital Subscriber Lines", IEEE Communications Magazine, pp. 25-32, Sept. 1989.

[15] Walters D.B, Line codes for metallic cable systems, INT. J. Electronics, Vol. 55, No. 1, pp 159-169, 1983.

[16] Buchner J.B, Ternary Line Codes, Philips Communications Review, Vol. 34, No. 2, pp. 72-87, June 1976.

[17] Ifeachor, Jervis. Digital Signal Processing, A Practical Approach. P186, P577-P609 Addison-Wesley.

[18] R.I. Damper. Introduction to Discrete-Time Signals and Systems. P165-166. Chapman & Hall. ISBN 0-412-47650-9.

[19] K. Fahy, E. Perez. Fast Fourier Transforms and Power Spectra in Labview. National Instruments application note number 040. Feb. 1993.

[20] Harris F.J. "On the use of Windows for Harmonic Analysis with the Discrete Fourier Transform", Proceedings of the IEEE, Vol.66, No. 1. Jan. 1978

[21] Saeed V. Vaseghi. Advanced Signal Processing and Digital Noise Reduction. P218-220. P220-221 Wiley. ISBN 0-471-95875-1.

[22] Proakis, Manolakis (1989), Digital signal processing: Principles, Algorithms and Applications. Prentice Hall.

[23] Bartlett ,M.S, (1948), "Smoothing Periodograms from Time Series with Continuous Spectra", Nature (London), vol. 161, pp.686-687.

[24] P.D Welsh (1967). "The use of the FFT for estimation of Power Spectra: A method based on time averaging over short modified periodograms". IEEE Transactions on Audio and Electroacoustics. vol. AU-15. no.2, pp.70-73.

[25] International Telecommunications Union (ITU-T), Recommendation I.430, Basic User-Network Interface – Layer 1 Specification, 1993.

[26] Harufusa Kondoh, Seiji Kozaki, Shinya Makino, Hiromi Notani, Fuminbu Hidani and Masao Nakaya, A Fully Integrated 6.25% Pull-in Range Digital PLL for ISDN Primary Rate Interface LSI, IEICE Transactions. Electronic, vol. E74-C, no.3, March 1992.

[27] Dr. Roland E. Best, Phase Locked Loops Theory Design and Applications, McGraw-Hill Book Company, ISBN 0-07-005050-3.

[28] International Telecommunications Union (ITU-T), Recommendation G.821, Error Performance of an International Connection forming part of an Integrated Services Digital Network, 1996.

[29] B.R. Wilkins, Testing Digital Circuits, Chapman and Hall, ISBN 0-412-38360-8.

[30] International Telecommunications Union (ITU-T), Recommendation O.150, Specifications of Measuring Equipment, 1996.

[31] Sol M. Shatz, Peter S. Kajka, Formal Modeling and Automated analysis of the LAPD Protocol, Journal of Computer Networks and ISDN Systems, vol 18 , pp 293-314, 1990.

[32] International Telecommunications Union (ITU-T), Recommendation Q.921, ISDN user-network Interface – Data Link Layer Specification. 1991.

[33] Bertrand Leigh, Michael Fingeroff, Douglas Morse, CPLDs Outshine HDLC Controllers in a Multichannel Design, Electronic Design Magazine, Nov 2, 1998.

[34] S Brown, R Francis, J Rose, Z Vranesic, Field-programmable Gate Arrays, Kluwer Academic Publications, 1992, ISBN 0-7923-9248-5.

[35] Actel Corporation, ProAsic(Plus) Flash Family FPGAs Product Brief, Oct 2003.

[36] Xilinx Inc, The Programmable Logic Data Book, 1994.

[37] Lucent Technologies, FPGA data book, 1998.

[38] Xilinx Inc, XC4000E and XC4000X Series Field-Programmable Gate Arrays, Data Sheet, 1997.

[39] Rohanimanesh, Eyogami, Poorali, Design of an ISDN Terminal, Proceedings of the 1995 IECON 21st, 1995, 2/2, pp 1598-1601.

[40] Thompson,G, ISDN Testing, Designing ATE to meet ISDN device test requirements, Part 1. Evaluation Engineering Magazine – March 1990.

[41] Brummer, Kostengunstig uber Zweidrahtleitungen, Elektronik, Nov 1992.

Report Number 1

A PSD Measurement Feature For Basic Rate ISDN.

Patrick Sugrue

Report number 1 is part of a portfolio of projects and is submitted in partial fulfilment of the requirements of the University of Glamorgan for the degree of Master of Philosophy.

15 – December – 2003

Contents

Appendix

List of Figures

List of tables

Abbreviations

ADC	Analogue to Digital Converter.
ADSL	Asymmetric Digital Subscriber Line
ANSI	American National Standard Institute.
ASSP	Application specific standard product.
ASIC	Application Specific Integrated Circuits
BERT	Bit Error Rate Test
DNL	Differential Non-Linearity Error.
DUT	Device Under Test.
DSL	Digital Subscriber Line
DSP	Digital Signal Processing
ENOB	Effective Number of Bits.
FPGA	Field Programmable Gate Array
HDSL	High Bit-Rate Digital Subscriber Line
IDSL	ISDN Digital Subscriber Line
INL	Integral Non-Linearity Error.
ISDN	Integrated Systems Digital Network
ITU-T	The Telecommunications Division of the International Telecommunications Union.
NEXT	Near End Crosstalk
PCB	Printed Circuit Board.
ppm	Parts Per Million
PRBS	Pseudo Random Binary Sequence
PSD	Power Spectrum Density

RAM	Random Access memory.
SDSL	Symmetrical Digital Subscriber Line
SNR	Signal to Noise Ratio.
VHDL	Very-High-Speed-Integrated-Circuit Hardware Description Language.
V.I.	Virtual Instrument.
2B1Q	Two Binary One Quaternary

Chapter 1

1 Introduction.

The close proximity of Digital Subscriber Line (DSL) signals within the same cable system, also called a cable binder can result in crosstalk levels that degrade performance. Near End Crosstalk (NEXT) has been recognised as the biggest problem and this is further exaggerated in DSL services that adopt a fully symmetrical data transmission system. The Spectrum Management Group, responsible for the publication of the T1E1.4 draft Spectrum Management Standard [1] has recognised that the practical way to limit crosstalk is to limit signal power applied to the telephone line. In the standard, the term spectrum management refers to processes that are intended to minimize potential for crosstalk interference and maximize the utility of the telephone line frequency spectrum. The T1E1.4 standard states that, "spectral compatibility is the capability of two transmission system technologies to coexist in the same cable and operate satisfactorily in the presence of crosstalk from each other". The standard applies to the Integrated Systems Digital Network (ISDN), ISDN Digital Subscriber Line (IDSL), High Bit-Rate Digital Subscriber Line (HDSL), Symmetrical Digital Subscriber Line (SDSL) and Asymmetric Digital Subscriber Line (ADSL).

The standard defines nine spectrum management classes into which each DSL technology may be categorised. Categorisation is dependant upon the operating data rate of the technology in question. Each spectrum management class defines the maximum signal power, deployment guidelines and PSD mask used in conformance measurements. BR-ISDN is categorised as spectrum management class 1. The PSD of the signal must be below a certain level as defined by a PSD mask. Violation of this

PSD mask may result in increased crosstalk interference with other signals within the cable binder.

The output power present in the output signal of an ISDN transmitter can be represented by a Power Spectrum Density (PSD) plot. A PSD mask defines a signal power limit that can be applied to the ISDN. Transmission power limits are placed on DSL technologies in order to reduce interference with other technologies sharing the same telephone cabling system. The T1E1.4 Draft Spectrum Management Standard defines a PSD mask for each of the DSL technologies. The standard also provides deployment guidelines for Telephone companies intending to mix different DSL technologies within the same telephone cabling system. The guidelines exclude the mixing of some DSL technologies within the same telephone cabling due to the resulting electrical interference.

The T1E1.4 Spectrum Management Standard has proposed a number of requirements that DSL transmission equipment should meet to ensure spectral compatibility. These are transverse balance requirements, longitudinal output voltage requirements, deployment guidelines and PSD limitations. DSL Equipment that meets these requirements is deemed, according to the draft standard to be spectrally compatible with other DSL technologies. The PSD requirements and the method adopted to determine compliance will be the subject of this project. The PSD requirements defined in the T1E1.4 draft standard have already been adopted by DSL equipment and chip set manufacturers and the measurement facilities to test compliance have been incorporated within test equipment. In order to facilitate the development of a PSD measurement feature, an ISDN signal generator that generates a known reference signal will be required. The development of this signal generator

will also be incorporated within the project work. Figure 1 identifies the two components in a typical PSD measurement set-up.

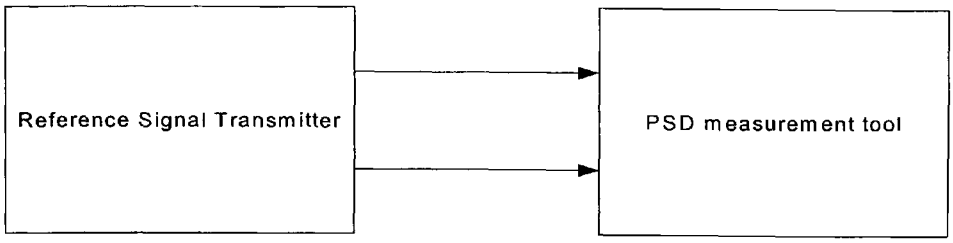


Figure 1 – The system block diagram.

1.1 The problem with current PSD measurement designs.

The design philosophy currently adopted by Test Equipment Designers is based on the utilisation of Application specific standard products (ASSP) to realise the hardware for test equipment. The space limitations imposed on designers due to the practical size that can be assumed by portable test equipment limits the number of electronic components employed in the design. This in turn limits the scope of functionality available to designers and usually leads to designs that address only one area of test. For example, should a PSD measurement facility be required, a specialised copper qualification tester is then required.

The copper qualification tester normally incorporates the PSD measurement feature along with other test features applicable to the deployment of DSL services on copper pairs. Should a test feature not associated with copper qualification be required, such as a Bit Error Rate Test (BERT) feature then a separate tester is usually required. It is unusual to find equipment incorporating both PSD measurement facilities and Bit Error Rate Test facilities. This serves as an example. However, the problem can be extended to the other areas of DSL testing. The lack of test feature

integration increases the range of equipment required by telephone companies along with the associated cost.

The DSL market is dynamic in nature and as new technologies are conceived and deployed, Test equipment manufacturers are expected to keep pace. The currently adopted design philosophy does not lend itself easily to product recycling. Test Equipment Manufacturers are usually required to redesign in order to address new advances in DSL technology. This produces a limited product life cycle for test Equipment. ASSPs are not easily modified without extensive cost, design effort, and time. Test Equipment Manufacturers are required to wait until ASSPs supporting new DSL features become available before test equipment design can be completed.

Currently deployed PSD measurement equipment is complex and usually requires an Engineer to operate effectively. The PSD measurement results are graphical and require some technical knowledge to interpret correctly. The development of a PSD feature that produces a pass or fail result would simplify testing. The PSD measurement feature could then be incorporated within portable testers that do not support large graphical displays.

Test equipment with the ability to integrate more test features as field upgrades when new DSL technologies evolve would offer distinct advantages over existing test equipment. Field Programmable Gate Array (FPGA) technology will be utilized to develop a reconfigurable hardware platform. The hardware platform is capable of integrating different test features for ISDN installation and testing applications. The Portfolio introduction chapter describes the hardware platform.

1.2 The project objectives.

The following is a list of objectives for project number 1.

1 To investigate the DSP techniques currently employed in spectrum analysis and apply these methods to PSD measurement.

2 To develop a configurable hardware platform utilising FPGAs to support PSD measurement at basic Rate ISDN data rates.

3 To implement a solution that is field upgradeable with the flexiblity to extend PSD measurement above the Basic Rate ISDN data rate without extensive redesign.

4 To develop an intelligent measurement system that does not rely on a skilled user to analyse the computed frequency spectrum and establish a pass or fail result.

1.3 Summary of achievements.

The main achievement was the integration of a PSD measurement facility within the reconfigurable hardware platform developed for the portfolio work. The hardware and software components required for the PSD measurement feature were successfully designed and implemented. The project involved the development of hardware components for a reference signal transmitter and the PSD measurement feature. The reference signal transmitter was designed to produce a Two Binary One Quaternary (2B1Q) reference signal. The reference signal was used to test and validate that the PSD measurement system operated correctly. The PSD measurement tool performed the analogue to digital conversion of the 2B1Q signal, storage of the

measured signal samples, and calculation of the PSD. The software was successfully developed to measure the PSD and compare the result with the PSD mask to produce a pass or fail indication. A successful PSD measurement was achieved using non-parametric techniques and the FPGA was successfully utilised to produce a flexible measurement system. It was shown how the modular design could be adapted to increase the measurement accuracy and how the system could be extended to measure the PSD for other data rates.

1.4 Report structure.

This report presents the hardware and software design involved in the development of the PSD measurement system. The contents of each chapter are outlined in the following section.

Chapter 2 presents the 2B1Q reference signal transmitter used to facilitate the PSD measurement process. The transmitter is composed of analogue and digital components. The analogue circuit design and functionality is presented and results of the analogue transmitter are shown. The Digital design process shows how FPGA technology is employed to interface to the analogue electronics and implement the pseudo-random binary sequence used in the measurement procedure.

Chapter 3 presents the 2B1Q signal measurement hardware. The measurement system is partitioned into analogue and digital components for explanation. The analogue discussion outlines the data-acquisition and associated analogue signal conditioning required to interface to the unit under test. The digital design discussion shows how the FPGA was used to facilitate the data measurement and implement the required interfacing between software and analogue hardware

Chapter 4 presents the software implementation. It is shown how the software controls the measurement and performs the PSD calculation.

Chapter 5 presents the results of the measurement process. Measurement results taken with different windows are shown and compared during the discussion.

Chapter 6 concludes the report by reviewing the objectives with respect to the achievements made. The novel method adapted to implement the test system is highlighted and the possibilities to provide other test facilities are discussed. Suggestions are made for future work and how the measurement integrity may be improved.

Chapter 2

2 The reference signal transmitter.

The reference signal transmitter provided the test signal used to design and validate the PSD measurement system. The transmitter was designed to generate a random 2B1Q signal. A functional block diagram in figure 2 shows the tasks that the system performs. The 2B1Q reference signal is required to be random in nature and the PRBS generator block provides a Pseudo Random Binary Sequence (PRBS) of length 2^{15}-1 [3]. This will provide a number of 32767 random bits before the pattern repeats. The 2B1Q encoder block converts the PRBS sequence into 2B1Q signal levels according to the 2B1Q coding rules presented in the portfolio overview. The pulse former filters the raw 2B1Q pulse amplitude signal and prepares it for telephone line transmission. The line driver block produces the differential signal at the required signal power for ISDN transmission. The system is connected to the line via an ISDN transformer.

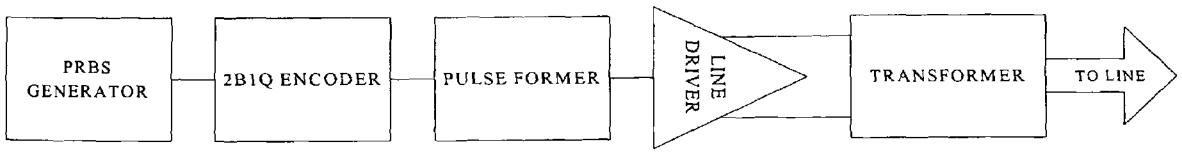


Figure 2 – Reference Signal Transmitter Block Diagram.

The block diagram presented in figure 3 is a more detailed top-level hardware representation, showing the analogue and digital circuit partitioning. The FPGA performs the PRBS generation, test sequence generation and interfaces to the analogue block. The analogue block generates a 2B1Q signal level for every two bits

received from the digital block. The required symbol filtering and line driver operation are all incorporated within the analogue block design. The inputs and outputs of the blocks are included on the diagram.

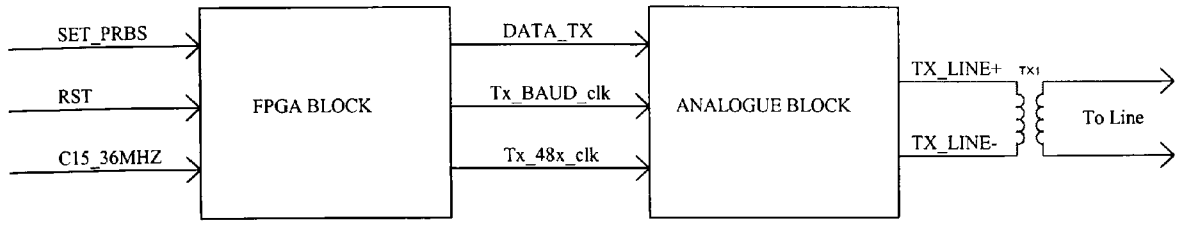


Figure 3 – Hardware Block Diagram for Transmitter.

2.1 The analogue circuit block.

The AFE1224 single pair analogue front end from Burr Brown was employed to provide the analogue functionality already mentioned. A data sheet is detailed in [2]. The block diagram of the AFE1224 is presented in figure 4. The 2B1Q reference signal transmitter is required to transmit only therefore the receiver section of the AFE1224 is redundant in this application.

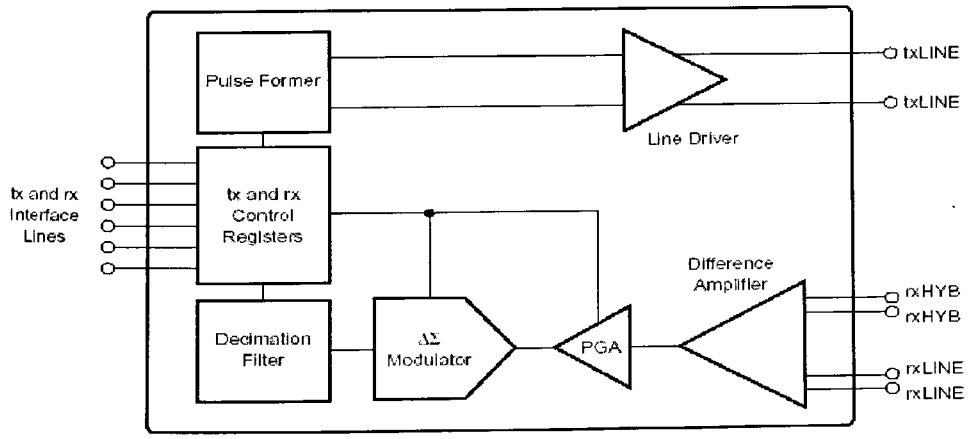


Figure 4 - Block diagram for the AFE1224 [2].

Analogue block inputs and outputs.

Three inputs to the AFE1224 are required to control the transmission circuitry and are described in the following.

TxBAUDclk: This clock is an input to the AFE1224. This is the baud rate of the transmitted data. For basic rate ISDN this is 80kHz to give a data rate of 160kbps.

Tx48xCLK: This clock is an input to the AFE1224. This is the oversampling clock used by the pulse former circuit. This clock should be 48 times the baud rate clock. For ISDN this is 3.84MHz (80kHz times 48).

Data_IN: This signal is an input to the AFE1224. Data is transmitted to the AFE1224 on the DATA_IN line and is formatted into a serial data frame that controls the operation of the chip. The data frame consists of 16 bits and is organized as that shown in figure 5. There are seven sub-sections to the data frame and figure 5 indicates how many bits are belonging to each sub-section. The sub-sections, tx_enable and tx_symbol are required by this application. The other sections can be set to logic 0. Table 1 presents the details of each sub-section within the data frame. The sub-section tx_enable is composed of one bit and is set to logic 1 to enable transmission. The sub-section tx_symbol is composed of two binary bits and these define the 2B1Q symbol to be transmitted. Two outputs from the AFE1224 are used to interface to the external ISDN signal transformer. These are described in the following.

TX_LINE+: Differential 2B1Q signal at 80kbaud.

TX_LINE-: Differential 2B1Q signal at 80kbaud.

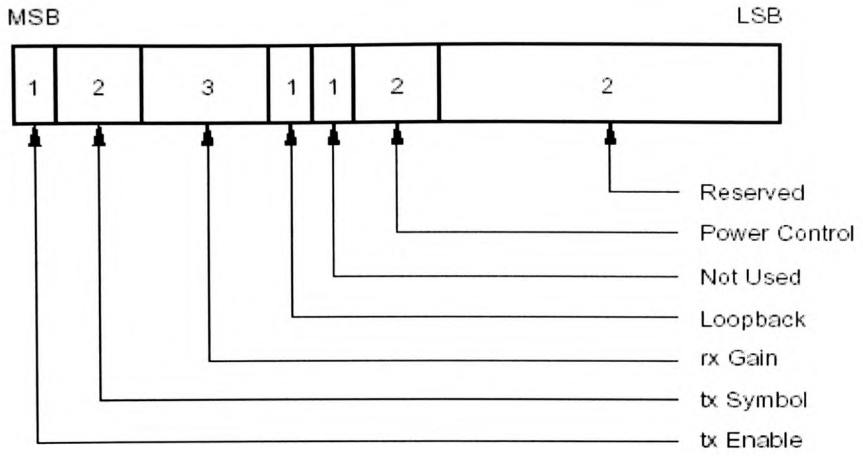


Figure 5 - Control data frame for AFE1224 [2].

BIT	DESCRIPTION	BIT STATE	OUTPUT STATE
15 (MSB)	tx Enable Signal	0 1	AFE transmits a 0 Symbol AFE transmits HDSL Symbol as defined by bits 14 and 13
14 and 13	tx Symbol Definition	00 01 11 10	–3 transmit symbol –1 transmit symbol +1 transmit symbol +3 transmit symbol
12 - 10	rx Gain Settings	000 001 010 011 100 101 110 111	rx gain in AFE 0dB rx gain in AFE 3dB rx gain in AFE 6dB rx gain in AFE 9dB rx gain in AFE 12dB rx gain in AFE reserved rx gain in AFE reserved rx gain in AFE reserved
9	Loopback Control	1 0	Go to loopback mode Normal Operation
8	Not Used		N/A
7-6	Power Control	00 01 10 11	Low speed, low power Medium Speed, power Normal Speed, power Normal Speed, power
5-0	Spare		N/A

Table 1 - Data frame format for AFE1224 [2].

The timing diagram for the analogue block interface.

The timing details, as outlined in the AFE1224 data sheet are presented in figure 6. Data should appear in the DATA_IN signal on the falling edge of the clock signal tx48Xclk. The AFE1224 reads data from the signal DATA_IN on the rising edge of the clock signal tx48Xclk. The falling edge of txBAUDclk signals the start of a frame. The position of the falling edge of txBAUDclk should be synchronous with the falling edge of tx48xCLK. The tx enable bit (MSB) is sent first followed by the two 2B1Q symbol definition bits. The other forty-five bits of the frame are set to logic 0. This will disable loopback and set the power level to low. The receiver gain is not important in this application.

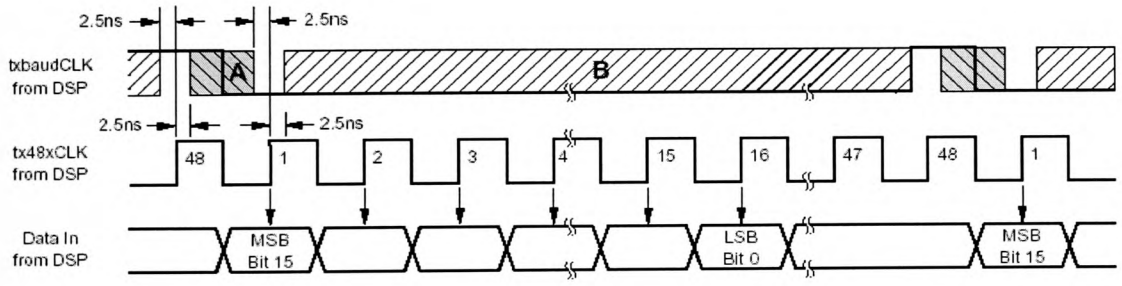


Figure 6 - Timing diagram for AFE1224 transmission control signals [2].

Implementation details for the AFE1224.

As the AFE1224 is a mixed signal integrated circuit careful attention to printed circuit board layout is required. The digital and analogue parts of the chip should be separated. In the actual implementation a separate PCB with a filtered power supply is used to populate the AFE1224 and associated analogue circuitry. Careful track layout was performed and a copper pour connected to ground was

placed underneath the analogue components. The copper pour occupies the whole of the PCBs bottom plane, with short breaks in the copper pour for component track connections and via holes. The AFE1224 circuit diagram is presented in figure 7. The diagram shows the inputs TXBAUDCLK, TX48XCLK and DATA_IN. Pins 26 (txLINE+) and 24 (txLINE-) are connected to the signal transformer via resistors R1 and R3.

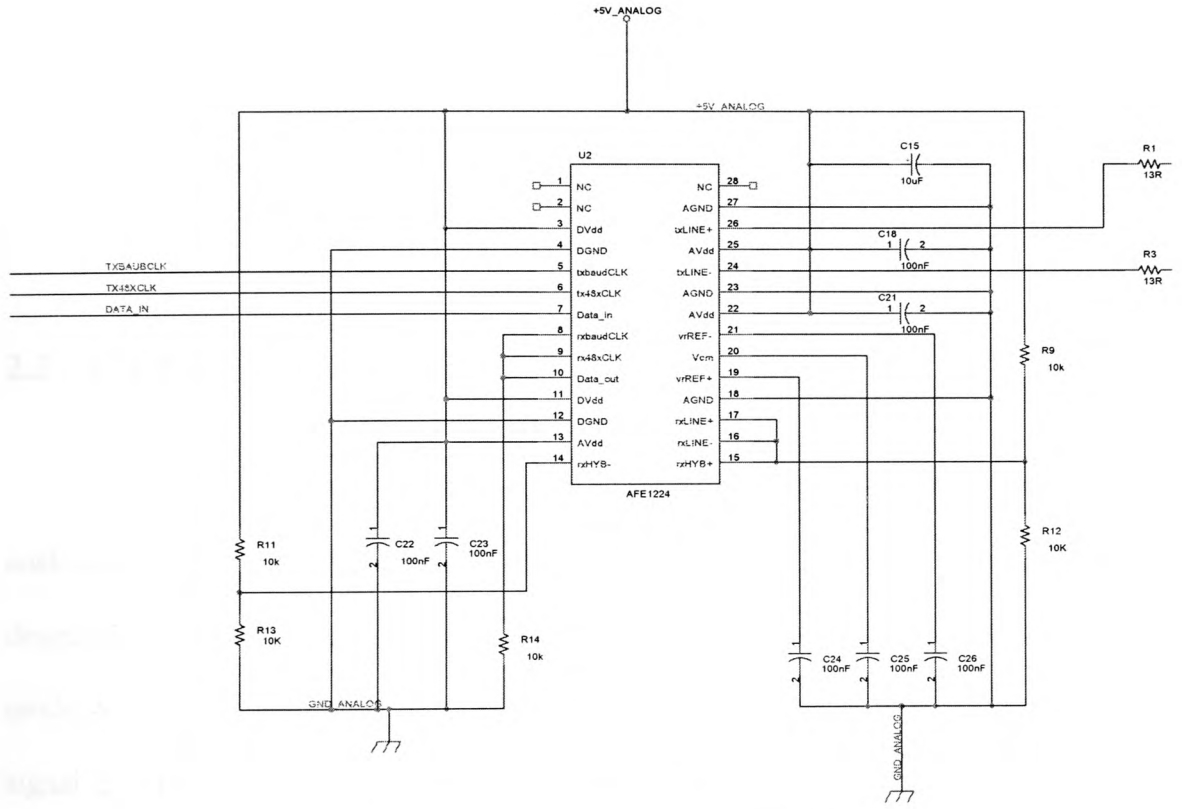


Figure 7 - Circuit diagram for AFE1224.

R1 and R3 in conjunction with C17 and C19 create a low-pass filter set to filter out signals above 556kHz. The reference signal is coupled to the telephone line via an ISDN transformer with a turns ratio of 1:1.6. The arrangement is shown in figure 8.

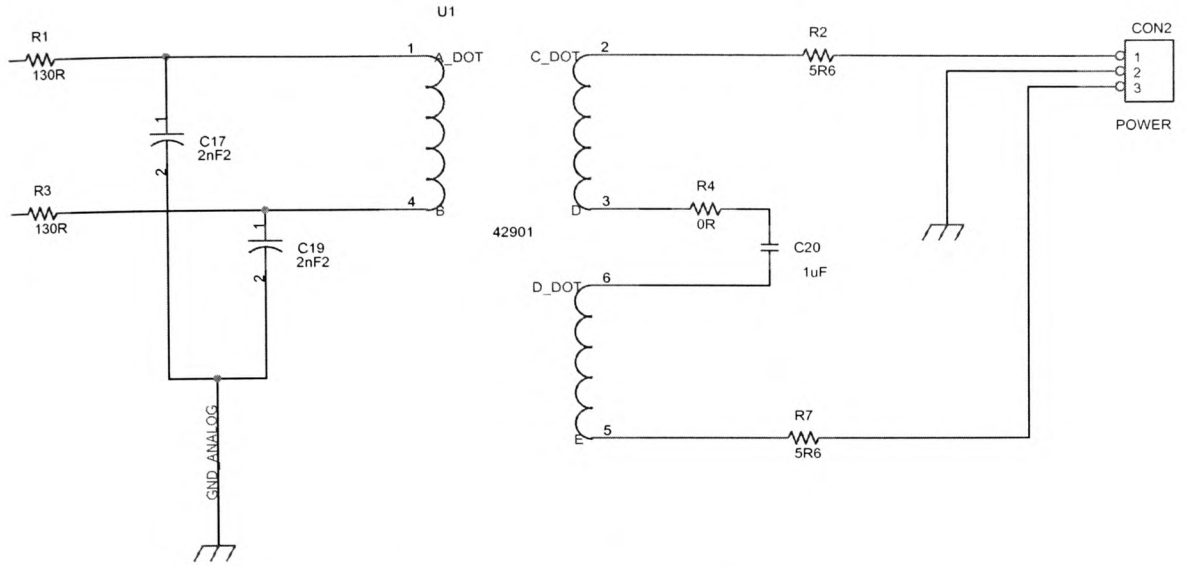


Figure 8 - Transformer coupling.

2.2 The FPGA circuit block.

The FPGA firmware architecture is required to provide an interface to the analogue block. A top-level block diagram is shown in figure 9. The FPGA was designed to support two modes of operation. A PRBS sequence is produced when mode A is active and a test sequence is generated when mode b is active. The input signal SELECT determines the mode of operation. The test sequence is used to validate the transmitted 2B1Q waveform from the AFE1224. The sequence is routed to the output OUT when the signal SELECT is set to logic high. The PRBS sequence is the default mode of operation and is active when the signal SELECT is set to logic low.

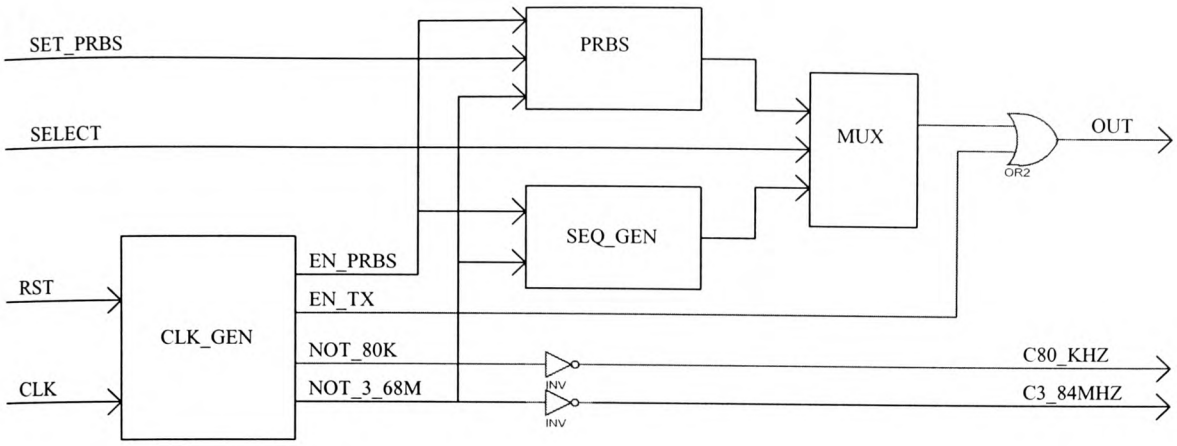


Figure 9 – FPGA Firmware Block Diagram

The FPGA block inputs and outputs.

The FPGA block inputs and outputs are described in the following.

SELECT: This is an input and determines the mode of operation. When set to logic low mode A is selected and the PRBS sequence is generated. When set to logic high mode B is selected and the test sequence is generated.

RST: This signal is an input and provides a synchronous reset to the system when set to logic high.

SET_PRBS: This is an input and resets the PRBS sequence when set to logic high.

CLK: This is and input and is the master clock for the system. The frequency is 15.36MHz +/- 100 parts per million (ppm).

OUT: This signal is an output and is connected to DATA_TX on the top level of hierarchy. The data frame required to control the analogue transmission is formatted and shifted out serially on this signal line.

C80_KHZ: This is the baud clock for the analogue block and produces the baud frequency of 80kHz at 50% duty cycle. This clock is applied to the signal txBAUDclk on the top level of hierarchy.

C3_84MHZ: This clock is connected to tx48Xclk on the top level of hierarchy.

The FPGA timing diagram.

The FPGA timing diagram is closely related to that presented in figure 6. The FPGA timing diagram is shown in figure 10. With reference to figure 6, the falling edge of the clock signal txBaudCLK (C80_KHZ) can occur anywhere in area A and the rising edge can occur anywhere in area B. Neither edge of C80_KHZ can occur within 4ns on either side of any rising edge of C3_84MHZ, according to the data sheet for the AFE1224 [2]. This can be achieved by deriving the signal NOT_C_80KHZ from the clock signal NOT_C3_84MHZ. NOT_C_80KHZ can be set high and low synchronously with the positive edge of NOT_C3_84MHZ. Both signals NOT_C3_84MHZ and NOT_C_80KHZ are then inverted to create C3_84MHZ and C_80KHZ before connection to the FPGA output. This will ensure that both edges of C_80KHZ will always be coincident with the negative edge of C3_84MHZ and always 130ns away from the nearest positive edge of C3_84MHZ. The count sequence is synchronous with the clock signal NOT_C3_84MHZ.

The clock signal C80_KHZ and the signals EN_TX and EN_PRBS can be derived from this count sequence. The signal EN_PRBS enables the PRBS generator or the test sequence generator for two NOT_C3_84MHZ clock cycles. The resulting two bits produced from this operation are stuffed onto the DATA_FRAME signal and sent to the analogue block. The signal DATA_FRAME is the serial data frame containing information to control the analogue block operation. The data frame is 16 bits long (labeled 15 to 0). Bit 15, the MSB, is sent first and the state of this bit when set high enables analogue transmission. Bits 14 and 13 contain the two-bit binary code that selects the 2B1Q symbol to be transmitted. For the remaining of the DATA_FRAME signal, a logic 0 is sent to the analogue block. Data must be valid 4ns before the rising edge of C3_84MHZ and must remain stable for at least 4ns after the

rising edge of C3_84MHZ. The DATA_FRAME signal is transmitted synchronously with the negative edge of C3_84MHZ (the oversampling clock for the AFE1224). This ensures that data is valid at a time equal to 130ns minus the FPGA routing delay and FPGA pad to output delay before the rising edge of C3_84MHZ. This timing consideration is shown in figure 11. The negative going edge of the baud clock C80_KHZ signals the beginning of the data frame.

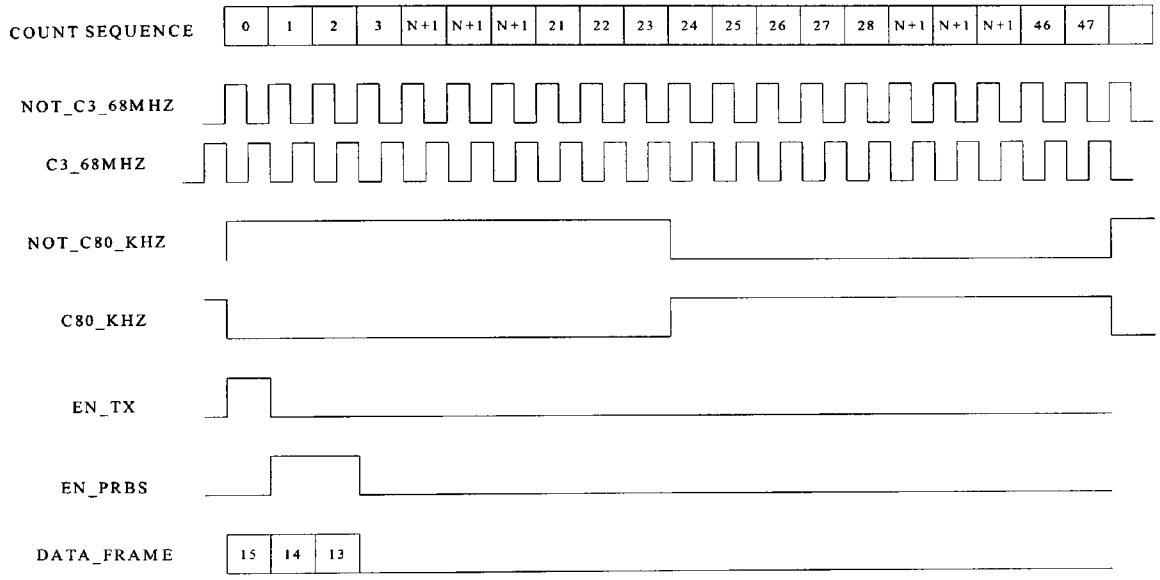


Figure 10 - Timing Diagram for the FPGA Block.

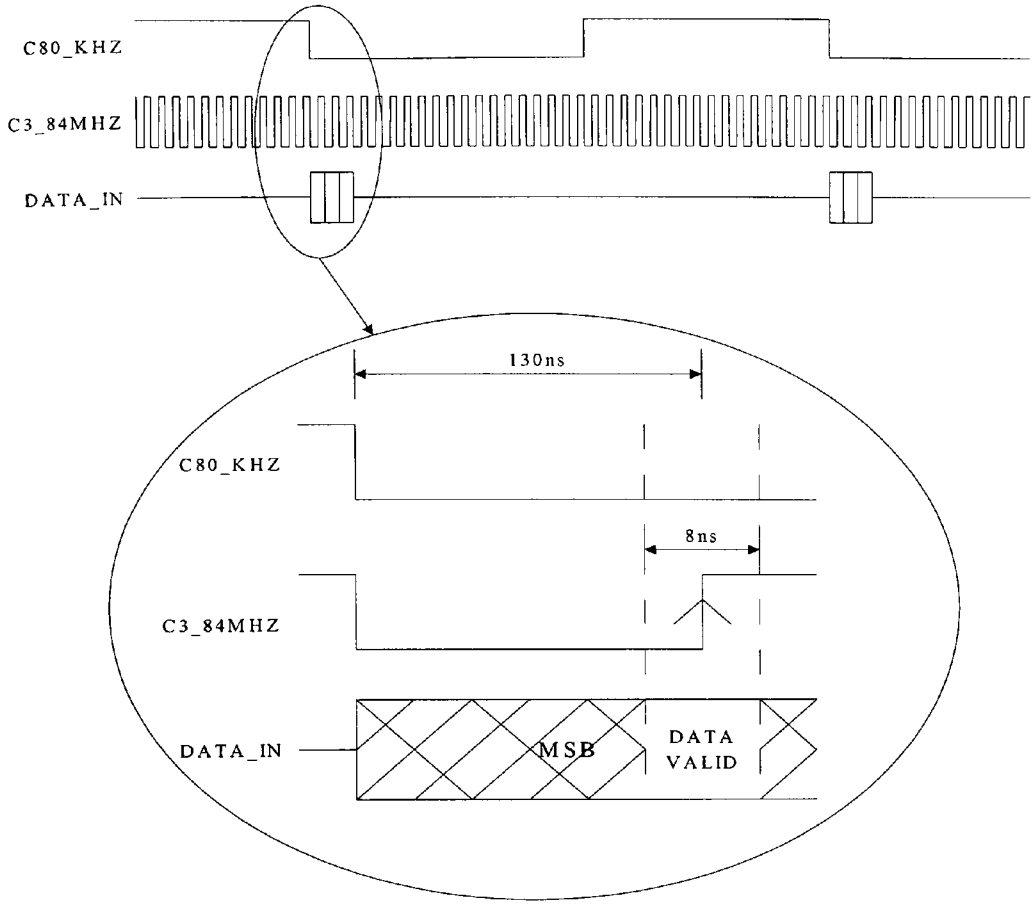


Figure 11 - Critical timing requirements for FPGA to analogue Interface.

FPGA sub-blocks

Each of the FPGA sub-blocks is implemented as digital circuits using the Very-High-Speed-Integrated-Circuit Hardware Description Language (VHDL). The PRBS block, the SEQ_GEN block, the MUX block and the CLK_GEN block are the lower level architectural circuit blocks in the FPGA hierarchical design. The functionality of each FPGA sub-block is described in the following.

The PRBS circuit block.

The PRBS circuit used is that defined in section 5.3 of the standard for measuring equipment O.150 [3] produced by the telecommunications Division of the

International Telecommunications Union (ITU-T). The PRBS sequence is 32767 bits long (2^{15}-1). The sequence is generated using a serial shift-register consisting of 15 D-type flip-flops were the 14^{th} and 15^{th} stages are XORed and fed back into the input of the first stage. The EN_PRBS signal is derived from the clock signal, NOT_3_68MHZ and enables the PRBS generator for two bit periods every baud period.

The SEQ_GEN circuit block.

A test sequence is required to verify the operation of the analogue 2B1Q transmission, as a random data stream is difficult to validate. A repetitive sequence consisting of +3, +1, -1, -3, -1, +1, +3 symbols was used to check the signal characteristics of the transmitter. The EN_PRBS signal is derived from the clock signal, NOT_3_68MHZ and enables the PRBS generator for two bit periods every baud period.

The MUX circuit block.

The MUX circuit block implements a digital multiplexer The PRBS sequence and the test sequence are inputs to the multiplexer. The logic state of the signal, SELECT determines which of the input sequences is routed through to the output. When SELECT is set to logic low the PRBS sequence is selected and when set to logic high the test sequence is selected.

The CLK_GEN circuit block.

The CLK_GEN circuit block has a lower level of hierarchy as shown in figure 12. The CLK_GEN circuit block is composed of the DIV_BY_4, CLK_COUNT, CLK_CNT_DECODE, and the BAUD_CLK_GEN circuit blocks. The operation of each block is explained in the following.

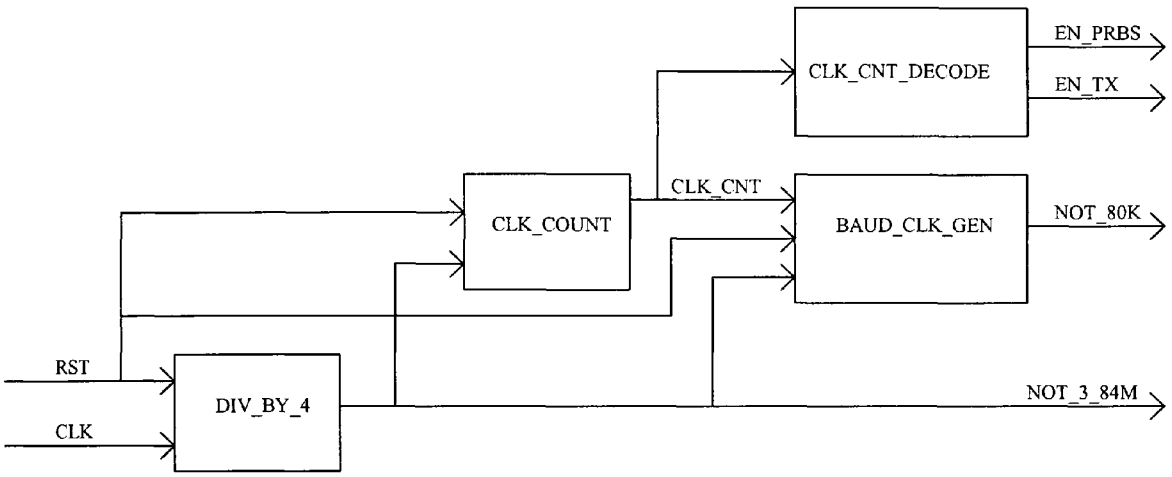


Figure 12 - FPGA Clock Generation Block Diagram.

The DIV_BY_4 block.

The DIV_BY_4 block divides the frequency of the input clock (CLK) by a factor of four to produce the clock signal NOT_3_84M running at a frequency of 3.84MHz (15.36MHz/4 = 3.84MHz) with 50% duty cycle. The input signal RST provides a synchronous reset to the blocks internal logic. The clock signal NOT_3_84M is used to clock both the CLK_CNT block and the BAUD_CLK_GEN block, therefore a clock buffer is placed on this signal. If a clock drives a multitude of flip-flops, it is possible that the active clock edge between one physical point of the FPGA may skew with respect to the same active edge at another physical point on the

chip. As a clock signal is routed physically across an FPGA it is possible that a time delay could occur between clock edges at different points on the FPGA due to different routing lengths. This is called clock skew. A clock buffer is employed to ensure low clock skew between different flip-flop clock inputs at different points on the FPGA. The clock buffer is an FPGA design component that is physically connected to each flip-flop clock input on the FPGA.

The CLK_COUNT block.

The CLK_COUNT block generates a count sequence from 0 to 47. The count sequence is represented by a six bit signal bus where CLK_CNT(0) is the least significant bit and CLK_CNT(5) is the most significant bit. The signal RST provides a synchronous reset to the blocks internal logic. The block is clocked with NOT_3_84MHZ therefore the period is as follows.

$$\frac{1}{3.84MHz} = 260ns$$

The count sequence will repeat at the baud rate of 80kHz, as 260ns times 48 clock cycles of C3_84MHZ is 12.5μs.

$$\frac{1}{12.5\mu s} = 80kHz = Baud...Rate$$

The AFE1224, used to generate the 2B1Q signal within the analogue block requires a baud clock (80kHz) and an oversampling clock (3.84MHz). A frame of data is sent to the analogue block every baud period (12.5μs). Therefore some means

of timing is required to synchronise the baud clock, the oversampling clock and the data frame sent to the analogue block. The six-bit binary count sequence, CLK_CNT, is used to facilitate synchronisation. The baud clock can be generated from the count sequence and the count sequence can be monitored to indicate when data should be sent to the analogue block.

The BAUD_CLK_GEN circuit block.

The BAUD_CLK_GEN circuit block generates the baud rate frequency at 80kHz and 50% duty cycle. The signal RST provides a synchronous reset to the blocks internal logic. The timing diagram for the BAUD_CLK_GEN block is shown in figure 13. The binary count from the CLK_COUNT circuit block (CLK_CNT) is represented in integer form by the signal labeled as count sequence on the diagram. The BAUD_CLK_GEN circuitry is clocked by the clock signal NOT_C3_84M. The count sequence is decoded and the signals BIT_23 and BIT_47 are produced synchronously with the clock signal NOT_C3_84MHZ. The signal BIT_23 resets the clock signal NOT_C80_KHZ low and the signal BIT_47 sets the clock signal NOT_C80_KHZ high.

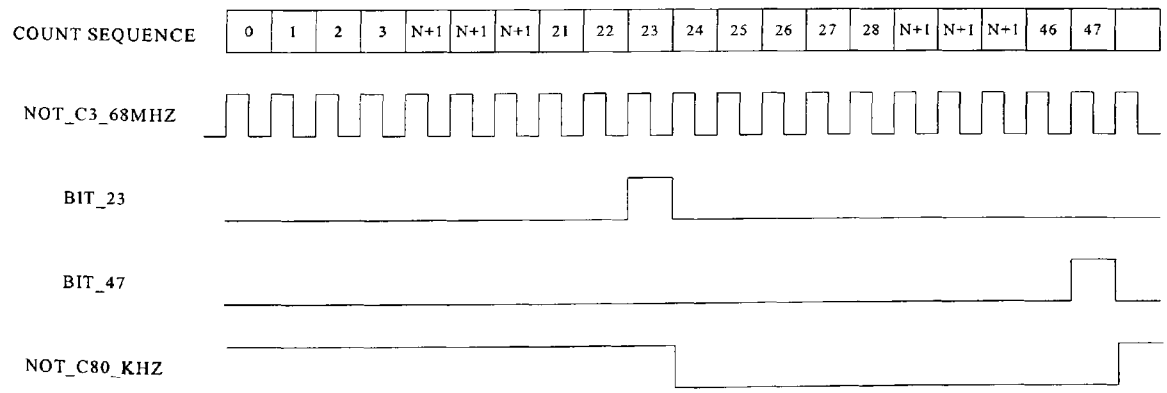


Figure 13 – Timing Diagram for the BAUD_CLK_GEN Block.

The CLK_CNT_DECODE circuit block.

The CLK_CNT_DECODE circuit block provides the enable signal EN_PRBS for the PRBS generator and the test sequence generator. The count sequence is decoded and when the count sequence is at state 0, the EN_TX is asserted high for one NOT_C3_84MHZ clock cycle. When the count sequence is at state 1 or 2, then the signal EN_PRBS is set high. This is represented graphically in the FPGA timing diagram in figure 10.

FPGA implementation Results

The place and route procedure reported a device utilization of 31 configurable logic blocks (CLBs) out of a total of 196 CLBs and the timing requirements were successfully achieved. A summary of the FPGA place and route process is presented in the following.

Number of External IOBs 6 out of 61 9%

Number of Global Buffer IOBs 1 out of 8 12%

Number of CLBs: 31 out of 196 15%

Total Latches: 0 out of 392 0%

Total CLB Flops: 41 out of 392 10%

4 input LUTs: 58 out of 392 14%

3 input LUTs: 5 out of 196 2%

Number of BUFGLSs 2 out of 8 25%

2.3 Verification and validation of the Reference Signal Generator.

The verification and validation of the Reference Signal Generator focused on the digital interface signals between the FPGA and the AFE1224 and on the resulting 2B1Q output signal from the AFE1224. The interface between the FPGA and the AFE1224 was extensively tested in the laboratory. The actual timing of the signals flowing between the FPGA and the AFE1224 were captured by a digital oscilloscope and are shown in figure 14 and figure 15. Figure 14 shows the oversampling clock (3.84MHz) on channel one and data on channel two. The positive going edge of the oversampling clock is coincident with the center of each data bit, as defined by the timing diagram in figure 6. The overshoot was caused by noise induced from the noisy digital circuitry into the clock track on the PCB. This was later rectified by cutting the clock track and using a wire, carefully placed so as to avoid the noisy areas of the PCB.

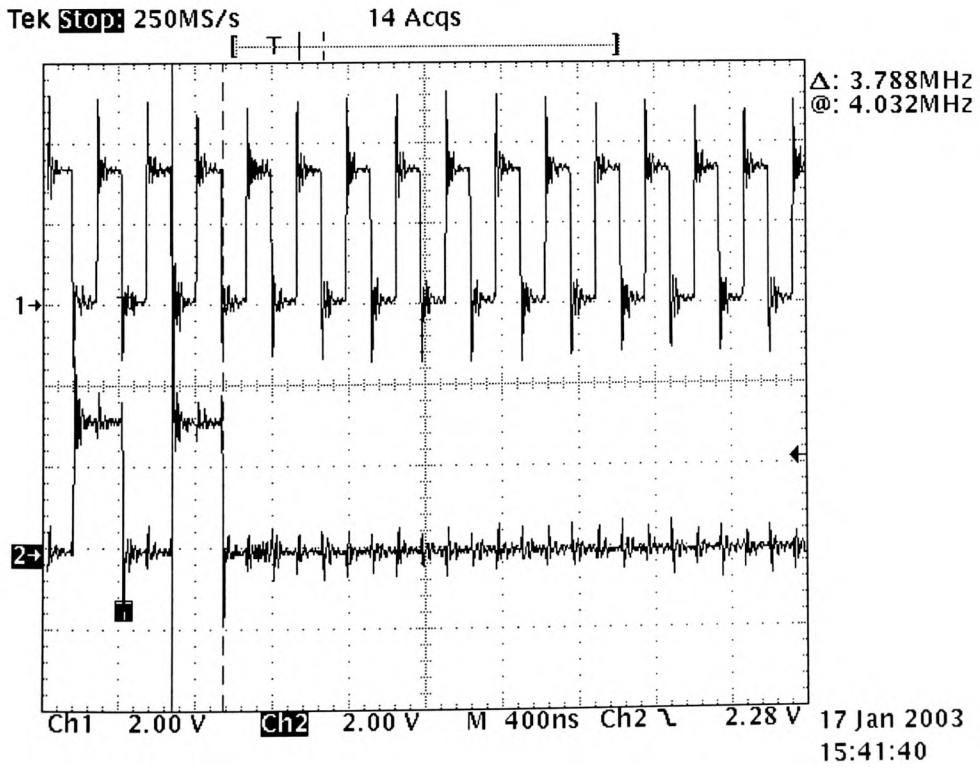


Figure 14 - The oversampling clock signal.

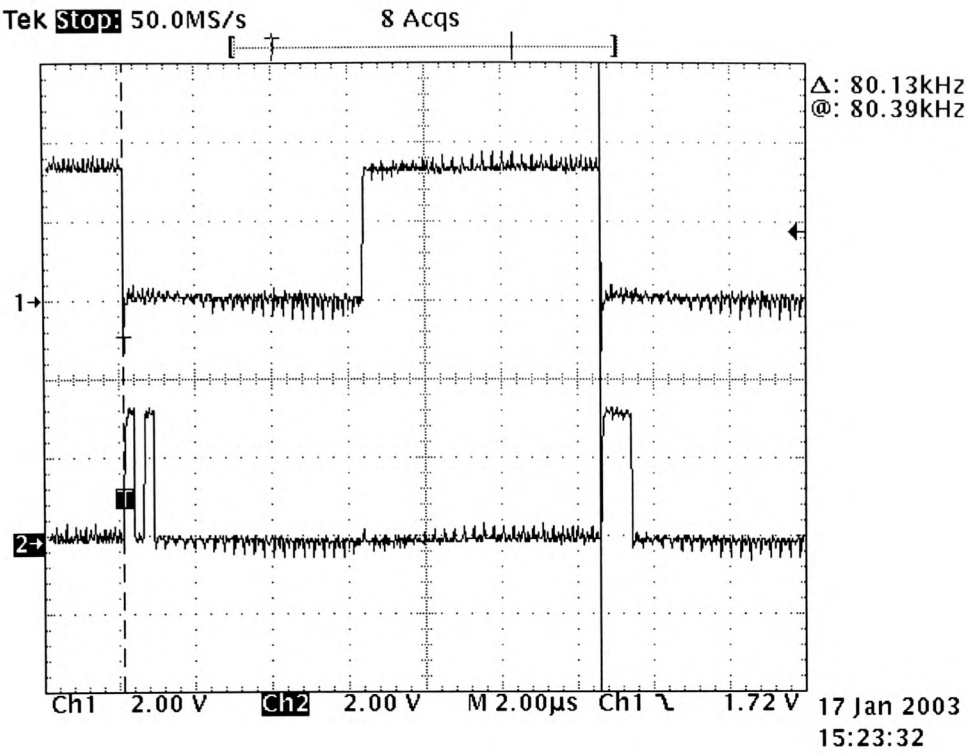


Figure 15 - The C80_KHZ baud CLK and data.

Figure 15 displays two periods of the baud clock signal C80_KHZ on channel one and two data frames on channel two. The negative edge of C80_KHZ is coincident with the beginning of each data frame. It can be seen that the data frames contain the bits 101 and 111 respectively. The oscilloscope cursers mark the period of the C80_KHZ signal and indicate a frequency of 80kHz as expected.

The FPGA is configured to transmit the test sequence consisting of +3, +1, -1, -3, -1, +1, and +3 quats repetitively. Figure 16 shows the resulting expected signal. The actual output from the reference signal generator is differential and not referenced to ground. A differential amplifier is required to convert the signal to a ground referenced single-ended signal for measurement. The signal amplitude is on the vertical axis and indicates quaternary symbols. The horizontal axis indicates time

from T1 to T6 in steps of 12.6μs (baud period). In this case the signal starts at T1 while a +1 symbol is transmitted. The next symbol to be transmitted is +3 and the signal rises over 12.6μs to reach the +3 symbol amplitude level. The signal will reach the maximum output voltage points when transmitting +3 and –3 symbols. The peak-to-peak voltage should not be more that 5.25V peak-to-peak. This value is obtained from the pulse mask defined by the American National Standard Institute (ANSI) in standard number T1.601 [4].

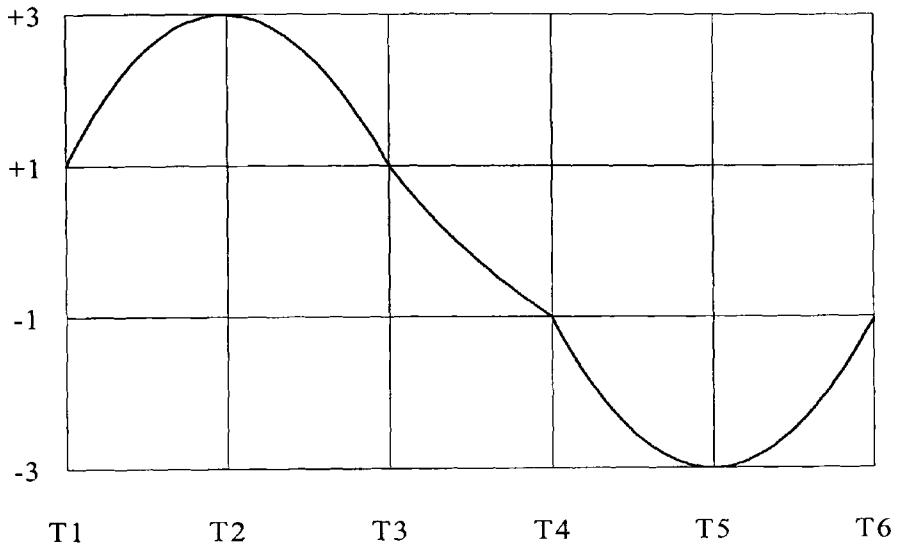


Figure 16 - The expected 2B1Q signal.

The INA121 instrumentation amplifier manufactured by Texas Instruments was used to perform the differential to single-ended operation. The amplifier is shown in figure 17 and a data sheet is referenced in [5]. The objective of the verification exercise is to validate the time domain signal from the 2B1Q reference signal transmitter. The measurements are made across a 135 ohm resistor at the transmitters output. The measurement results are discussed in the following. Figure 18 shows the differential signal at the output and figure 19 shows the single-ended signal at the output of the INA121 differential amplifier. The peak-to-peak

signal is less than 5.25V. The transmitted signal is as expected and time between signals is 12.6µs. This can be seen in figure 20, where the time between a +1 symbol and –1 symbol is marked using the oscilloscope cursers. Figure 21 shows the repetitive +1, +3, +1, -1, -3, and -1 voltage levels again and here the cursers mark out the time between the +3 level and the –3 level. The measured time is 38µs as expected. An example of a random 2B1Q signal can be seen in figure 22.

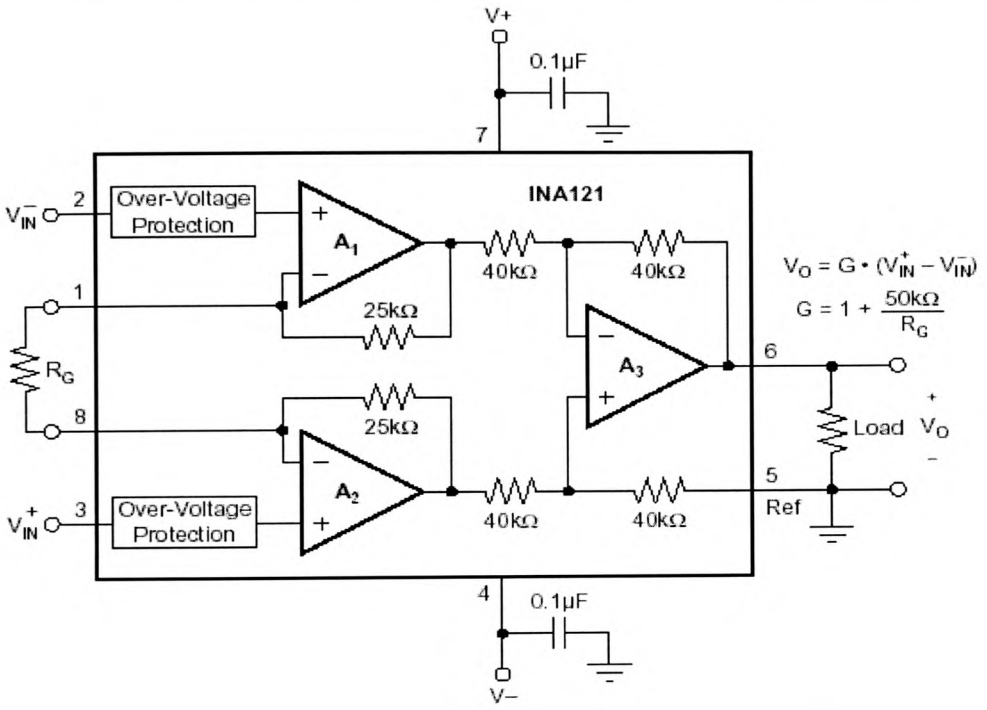


Figure 17 - The INA121 instrumentation amplifier [5].

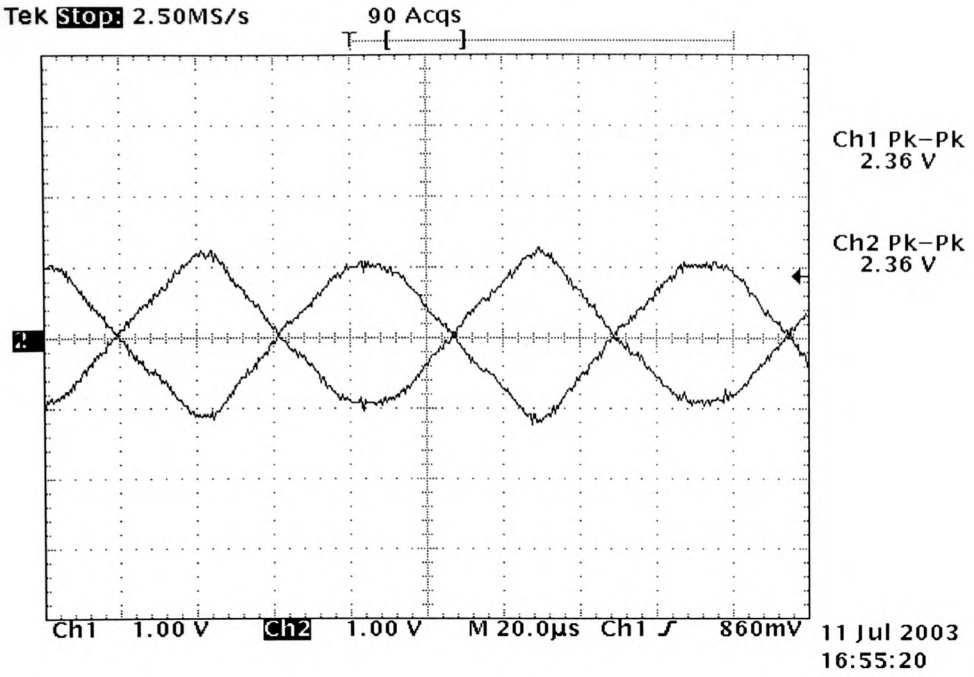


Figure 18 - Differential test signal.

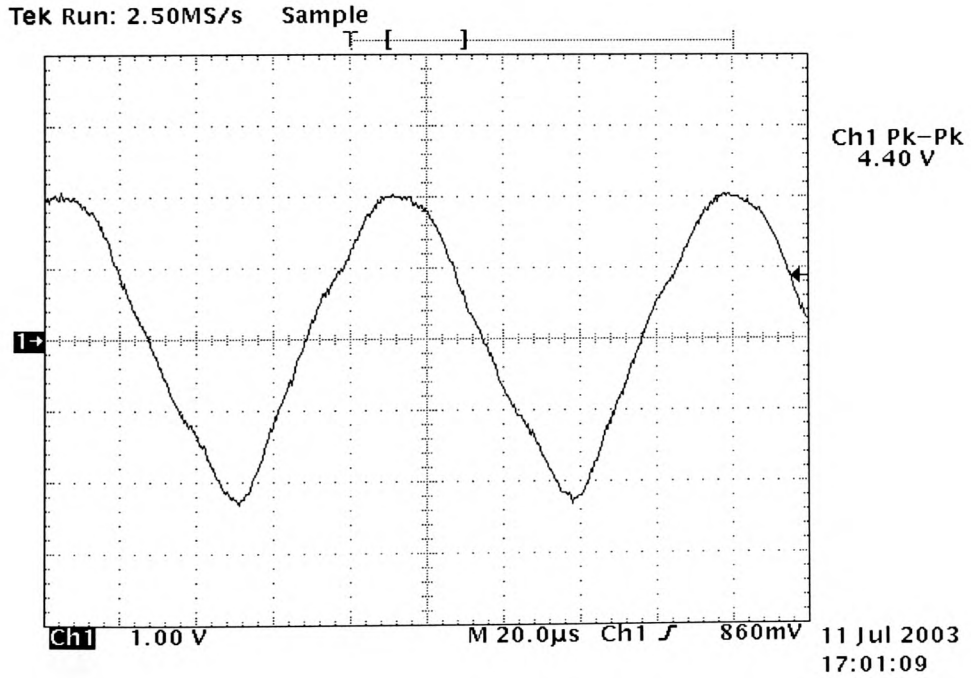


Figure 19 - Single-ended test signal.

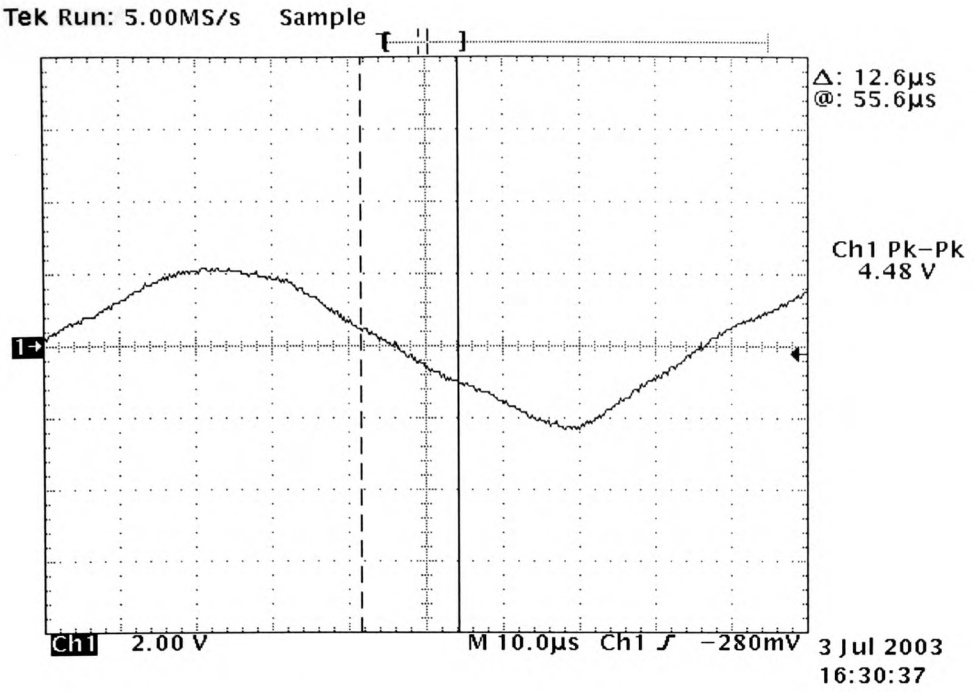


Figure 20 - The time duration for one 2B1Q symbol.

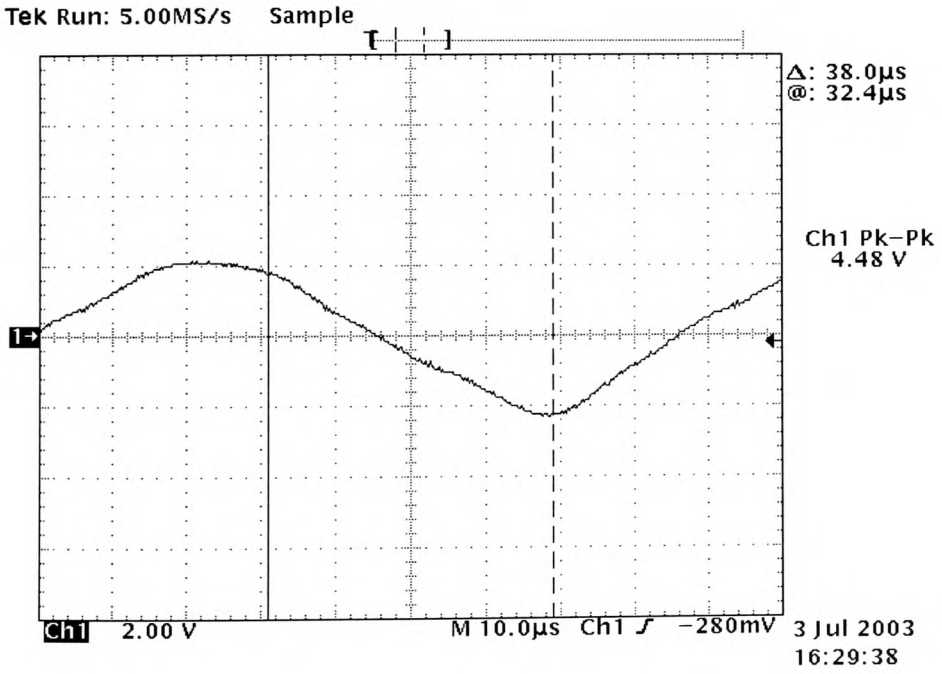


Figure 21 - The time duration of three 2B1Q symbols.

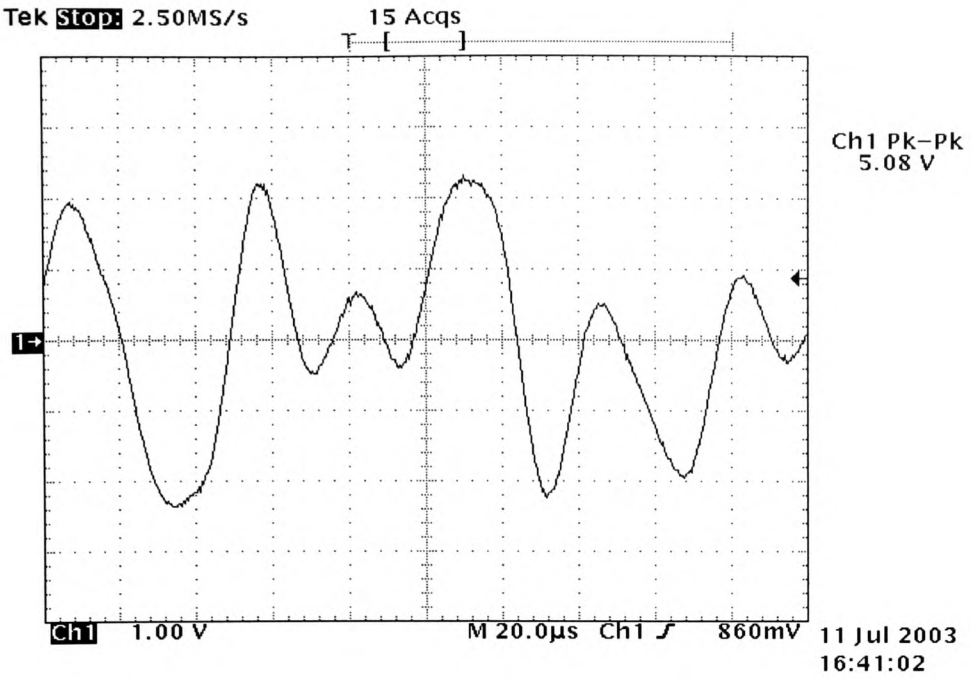


Figure 22 - The random 2B1Q signal.

Chapter 3.

3 The reference signal receiver.

The reference signal receiver monitors the signal transmitted from the Device Under Test (DUT) and calculates the signals PSD across a 135ohm resistor. In this project the Reference signal transmitter simulated the DUT. The PSD is then compared to a PSD mask and a pass or fail indication is made. The system employs software and hardware to complete the task. The software runs on a computer and the computer interfaces to the hardware measurement system via a National Instruments DAQ card. The block diagram for the measurement setup is shown in figure 23. A photograph of the laboratory test setup is shown in figure 24. The system shows a Notebook Computer with the National Instruments DAQ card inserted. The DAC card is connected via the ribbon cable and connector block to the FPGA on the larger of the two printed circuit boards (PCB). The FPGA interfaces to the analogue measurement hardware, which is located on the smaller of the two PCBs.

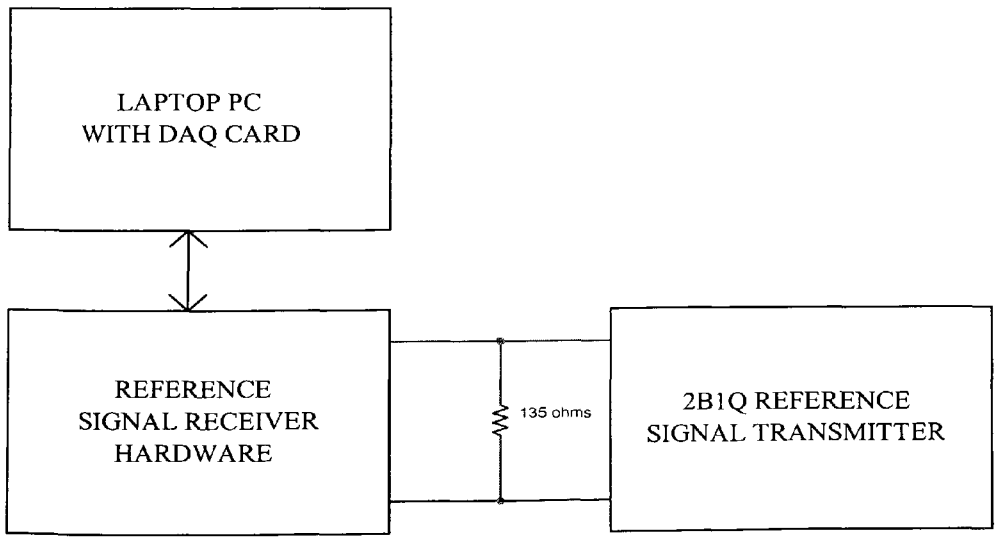


Figure 23 - Block diagram of measurement setup.

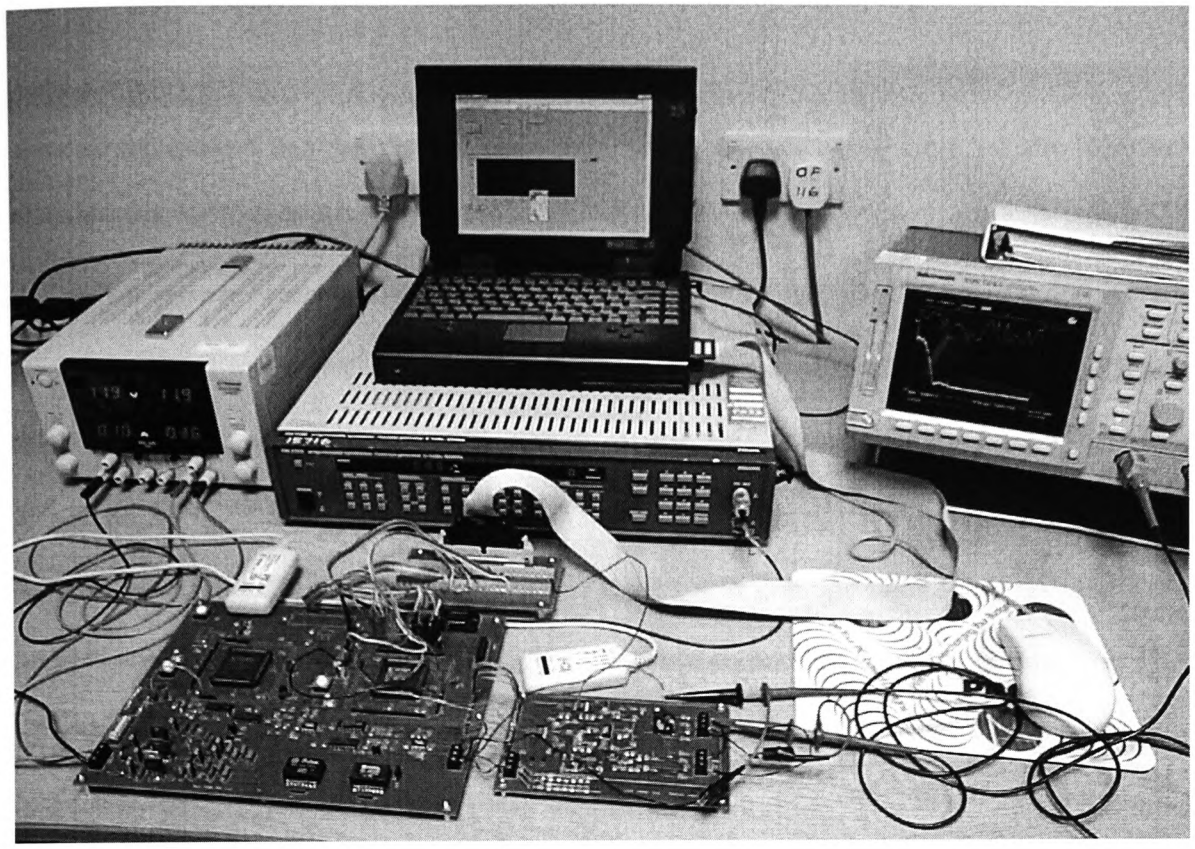


Figure 24 - Photo of test setup.

The hardware can be partitioned into DAQ CARD, FPGA and ANALOGUE RX blocks as shown in figure 25. In the following sections each hardware component will be explained.

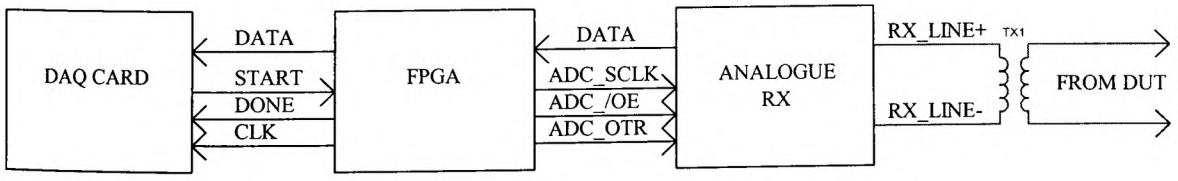


Figure 25 - Block diagram of the reference signal receiver hardware.

3.1 The DAQ Card Block.

The National Instruments DAQ CARD-1200 was used in this application and a data sheet is referenced in [6]. The software interfaces with the DAQ card via the National Instruments NI-DAQ driver software. The signals DATA, START, DONE and CLK defined in figure 25 are software control signals and are handled automatically by the NI-DAQ driver software. The Labview software programme was written to initiate a measurement, retrieve the data samples from the FPGA and calculate the PSD. The DAQ cards digital I/O facilities are used in this application. The software configures digital ports A and B (both 8 bit ports) as inputs and concatenates them to produce a 16 bit data bus. The signal DATA in figure 25 represents this 16 bit data bus. Port C is used to provide hand shaking between the software and the FPGA. The software configures Port C to operate in strobed input mode (mode 1). The DAC card generates the signal START. This signal is derived from bit 6 on port C, denoted PC3 and is physically accessed at pin 36 on the cards connector. A low to high transition on this line initiates a data acquisition. The data acquisition process captures 512 samples. Each sample is 16 bits. A sample is composed of measurement data (12 bits) and status information (4 bits). The status information is used to inform the software that an error has occurred and the measured data should be discarded. The 4-bit status signal indicates all zeros when the measurement is valid and all ones when the data is in error. The signal DONE is generated by the FPGA when the data acquisition process is complete. DONE is connected to PC7 on port C (pin 37 on the card). This indicates that there are 512 samples in the FPGAs RAM and data retrieval from the FPGA may begin. The software initiates data retrieval by a high to low transition on the START signal line. The FPGA generates the signal CLK and places consecutive data samples on the bus synchronously with the positive edge of the signal CLK. The

CLK signal is connected to the strobe pin on the card (PC2 or STBb, pin 32) and the negative edge of CLK latches data from the data-bus into the DAQ cards memory. When the DAQ card receives 512 clock edges from the FPGA the process is complete. At this point in time the buffer in the DAC card is full of measurement data and the Labview software programme can retrieve the data in its own time. The software may complete multiple data acquisitions by repeating this process.

3.2 The FPGA Block.

The FPGA block performs two tasks. Each task is initiated under software control. The first task is to interface to the Analogue to Digital Converter (ADC) in the analogue block and perform a data-acquisition (data-acquisition mode). One data-acquisition consists of 512 samples resolved to 12 bits. The samples are stored in RAM and an indication is sent to the software when the operation is complete. The FPGA signals used in the DAC card interface are START, DONE and CLK and have already been explained. The signals used to facilitate the ADC interface in the analogue block are ADC_CLK, ADC_/OE, and ADC_OTR. The signal, ADC_CLK, is used to clock the ADC. The signal, ADC_/OE, is an active low signal to enable the ADC. The ADC produces the signal, ADC_OTR, and this signal is set high by the ADC if the measured analogue signal exceeds the input range of the ADC. The signal, DIN, is the 12-bit data bus from the ADC.

The second task that the FPGA performs is to interface to the National instruments DAC card, sequentially address the 512kbytes of data in RAM and send it to the DAC card synchronously with the clock signal DAC_CLK (data upload mode). The first step in the design process was to define a block diagram and to identify the

required components in the design. The FPGA block diagram is shown in figure 26. The ADC/DAQ Card Interface block contains the electronics to interface to the ADC and DAQ card and to control the RAM read and write operations. The RAM address counter provides the address bus for RAM. The FPGA RAM block contains the static RAM for the system. The clock divider generates the master clock for the FPGA from the external 15.36MHz crystal oscillator. The design will be presented in a top-down design fashion and the details for each of the individual blocks will be presented in the following sections.

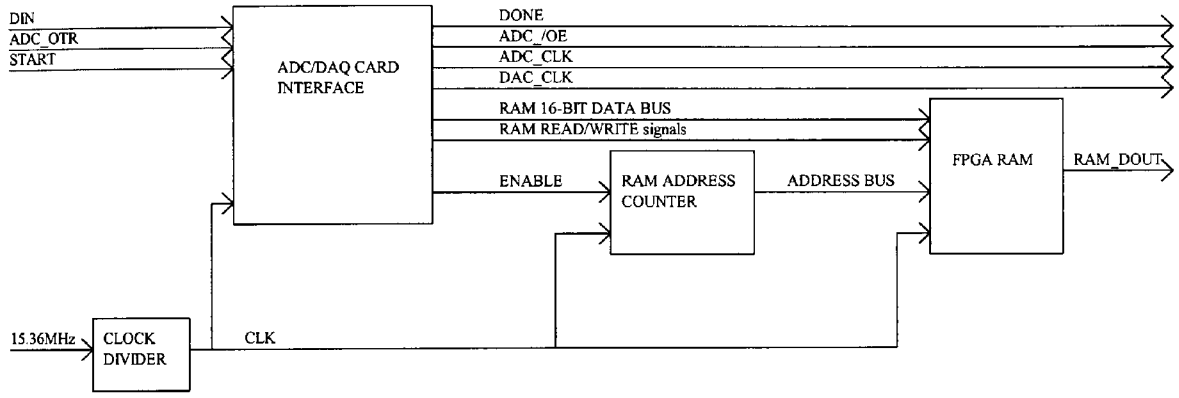


Figure 26 - The FPGA firmware block diagram.

The FPGA timing diagram.

The timing diagram was the second step in the design process and it defines the timing relationship between the signals in the FPGA system. The system timing will be considered for each of the two tasks performed by the FPGA. In data-acquisition mode the FPGA takes 512 measurement samples and stores them in RAM. During data-upload mode the FPGA sends the 512 samples to the DAQ card.

Following a reset to the FPGA, a low to high transition on the START signal line runs the process. The process starts in data-acquisition mode and once complete asserts the DONE signal high. The timing diagram is shown in figure 27. The START

signal may occur asynchronously with the master clock (CLK). START is synchronised to the master clock with an edge-detector driven by the master clock to produce the signal START_EDGE. Once the data-acquisition mode is complete, a consecutive low to high transition on the START signal runs the data-upload mode. Should it become necessary to run a second data-acquisition task after the first is complete a reset must be applied to the FPGA. In normal operation this should not be necessary.

The states that exist during the data-acquisition mode are defined in the timing diagram. The process decides on the next state by considering both the current active state and the current state of the input signals. States S1 through S4 are assigned to data-acquisition mode and states S5 through S11 are assigned to the data data-upload mode. Eleven states are used in total. Throughout this project one-hot encoding is employed. This technique uses one flip-flop to represent each state. With eleven flop-flops to total number of possible states is 2048 (2^{11}). At the state machine implementation stage the state decoding logic is designed so that unused states are not entered erroneously. In data-acquisition mode the FPGA interfaces directly with the ADC in the analogue block. Here the signals ADC_CLK and ADC_/OE are outputs from the FPGA. When ADC_/OE is driven low and the clock signal ADC_CLK (the inverse of CLK) is enabled, the ADC samples its analogue input signal. There is a data latency of eight clock cycles from the time a measurement begins to when valid data is available. Timing is achieved by driving an address counter with CLK. The address counter counts from 0 to 511 and is enabled with the signal EN_CNT. A data latency time-out is achieved by monitoring the count sequence RAM_ADDRESS and when the count is 7, data can be read from the ADC. One the data latency time-out has passed the count sequence RAM_ADDRESS is reset with the signal, RST_CNT.

The count sequence then starts from 0 and data is read from the ADC and stored in RAM until the count sequence reads 511.

The ADC samples data on the rising edge of ADC_CLK and data is valid on the negative edge of ADC_CLK. The signal CLK is the master clock for the system and all operations occur synchronously with it. ADC_CLK is the inverse of CLK and is enabled when EN_ADC_CLK is asserted high. Measurement data from the ADC is latched into the FPGA on the positive edge of CLK. Data is latched into the FPGA during state S3 and the RAM write enable signal (WE_RAM), loads the data into each consecutive address in RAM. Once all 512 locations of RAM have been loaded with data state S4 becomes active for one clock cycle and sets the DONE signal high. The process then moves to state S5 where it will remain until the next low to high transition of the START signal is identified by the FPGA. When the next start condition is identified, the process leaves data-acquisition mode and commences data-upload mode.

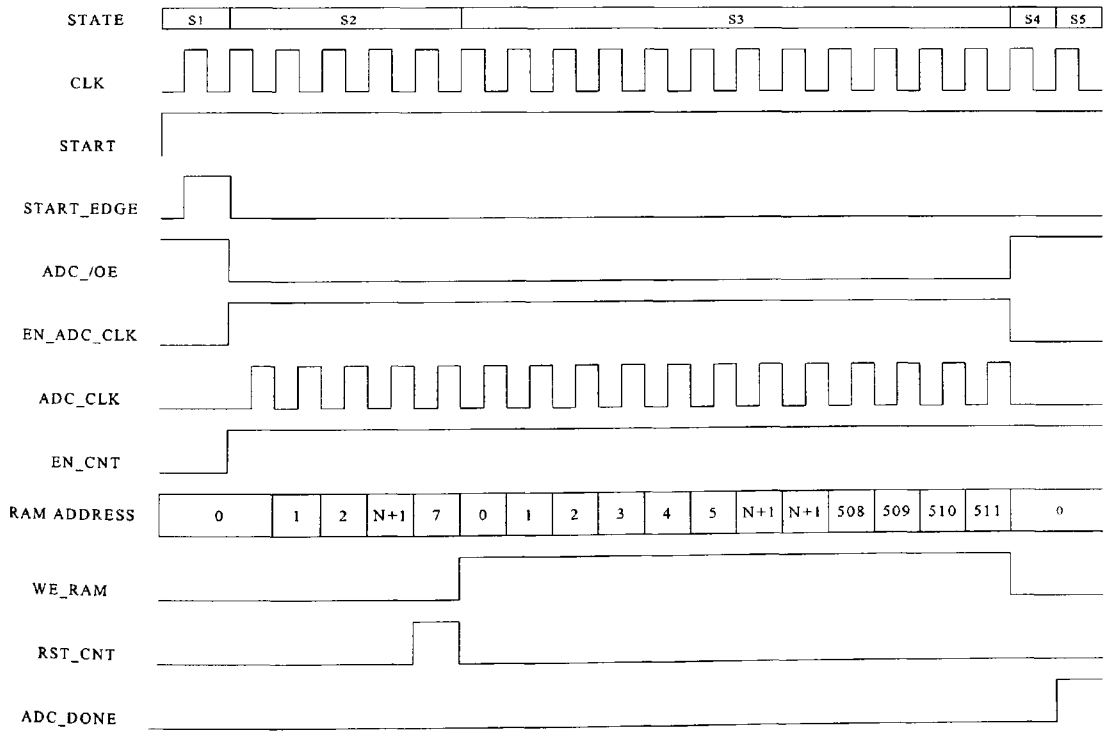


Figure 27 - The timing diagram for data-acquisition mode.

Data upload mode can be divided into two operations. The first is the DAQ timeout and the second is the RAM data upload operation. The DAQ timeout is required to allow for NI-DAQ software configuration before uploading RAM data. The timing diagram for DAQ timeout operation is presented in figure 28. When the signal START_EDGE goes high during state S5 the timeout starts. State S6 becomes active and the signal, EN_DAC_CLK, enables the clock signal, DAC_CLK. DAC_CLK is an output from the FPGA to the DAQ card. The DAQ card uses the DAC_CLK signal to capture data from the data bus. DAC_CLK is derived synchronously from the master clock signal. The signal, DAC_CLK_EDGE, is produced on each consecutive positive edge of DAC_CLK. DAC_CLK_EDGE lasts for one clock cycle of the master clock signal.

The address counter is used to facilitate timing and is clocked with the master clock signal. The count enable input of the address counter is connected to DAC_CLK_EDGE. The address counter is now only advanced on every positive edge of DAC_CLK. Employing a clock enable in the logic design allows different parts of the design to be clocked at different rates while using one clock signal for all logic in the design. In this case the master clock signal (CLK) runs at a frequency of 512kHz. The signal DAC_CLK runs at a frequency of 64Hz and is derived by dividing CLK by 8000. There are 8000 clock cycles in one period of DAC_CLK and the method of representation in figure 28 and 29 only shown 8 cycles of CLK for every DAC_CLK period. Although the time relationship between signals may not be conveyed accurately, the relationship of the signals with respect to their edges is clearly defined. The time-out lasts for the duration of state S7. When the address counter sequence

ADD_COUNT reaches 255 the next state, S8 is entered. The DAQ time-out operation is now complete and the data upload mode can commence on the next positive edge of DAC_CLK.

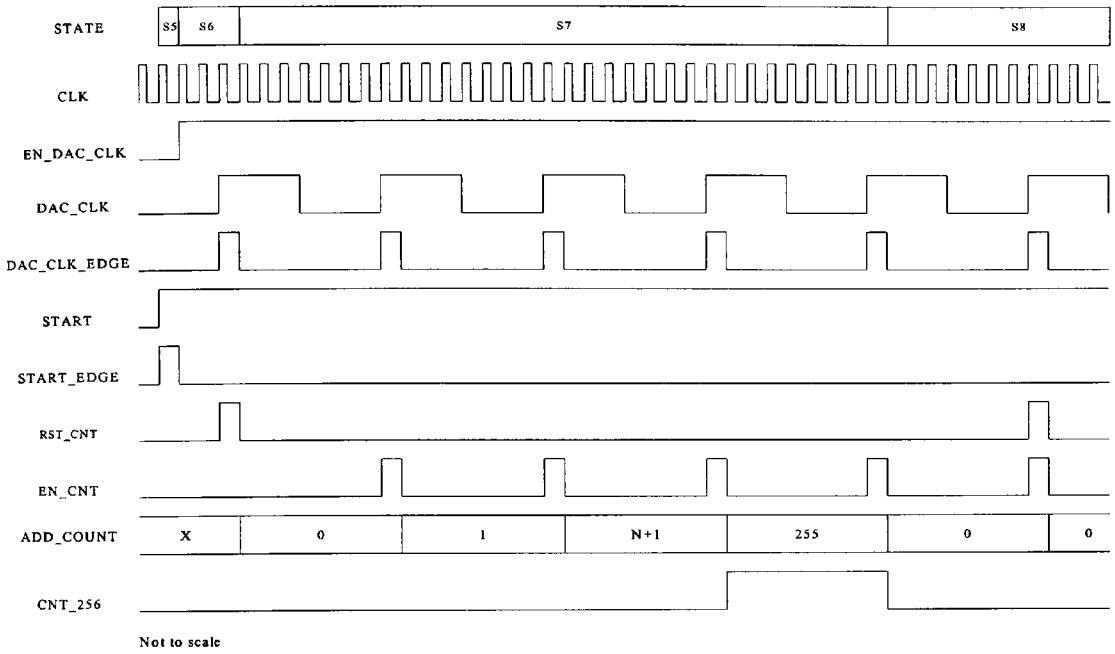


Figure 28 - The timing diagram for data-upload mode – Part A.

During the DAQ data upload mode the signal DAC_CLK is enabled as an output to the DAQ card. The FPGA reads data from RAM and places it on the 16 bit data bus DATA synchronously with DAC_CLK. This procedure is repeated until the address counter sequence reads 511. At this point in time, the state sequence moves from S9 to S10 and one more DAC_CLK signal is sent to the DAQ card before the state sequence makes a transition to S11. The next state is S1 and the system waits for another low to high transition of the start signal START before the data-acquisition mode is entered again.

The 64kHz clock frequency chosen for the data upload mode ensures that no timing problems are encountered. The data-acquisition mode lasts for 1ms (1.95μs

times 512 samples) and the data upload mode lasts for 8 seconds (15.6ms times 512). Once the software initiates a data acquisition the NI-DAQ driver takes between 3 and 4 seconds to configure the card for strobe mode. Due to FPGA design size restrictions the same counter was utilized for both data-acquisition mode and data up-load modes. During the DAQ timeout operation, the time for the address counter to sequence from 0 to 255 is 4 seconds (0.0156ms times 256). This timing is adequate to validate the operation of the system. It is clear that such long data upload times would not be practical in practice. The current hardware implementation has a PIC microcontroller wired to the FPGA and an RS232 connection for computer interfacing. This facility would allow for faster data upload times but solution was not developed during the exercise.

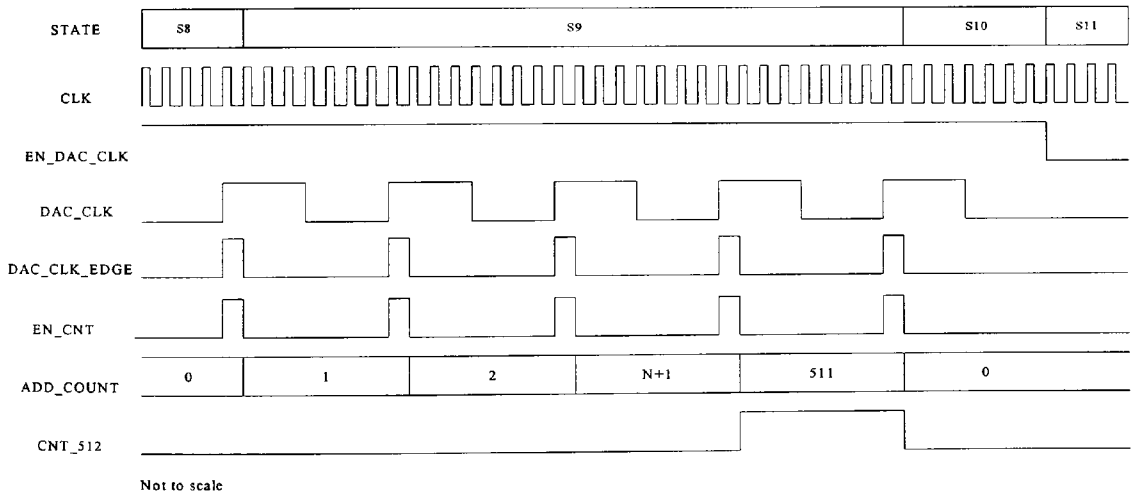


Figure 29 - The timing diagram for data-upload mode - Part B

The FPGA firmware design.

The FPGA block diagram is shown in figure 26 is a top-level representation. Reference here will be made to the implementation specific block diagram in appendix a, figure 1. The block diagram is composed of four circuit blocks, the ADS807 block (corresponds to the ADC/DAQ card interface block in figure 26), the RAM block, RAM_ADD and the MCLK_DIV block. A 15.36MHz crystal is employed as the master clock for the FPGA and enters on pin 13, labeled clock_pad. The 15.36MHz crystal is buffered with a dedicated clock buffer before being used to drive the FPGA logic. The signal RST resets the FPGA logic.

The MCLK_DIV block.

The MCLK_DIV block divides the crystal frequency down by a factor of thirty to give a clock frequency of 512KHz. This is the master clock frequency for the FPGA. And is buffered with a global clock buffer to drive to other blocks in the design. The BUFG buffer shown appendix A, figure 1 is directly connected to every clocked element on the FPGA. This provides a low skew clock signal to all the blocks in the design and facilitates timing closure. The inputs are C15_36MHZ and RST. RST is a synchronous reset and once asserted the output CLK goes low after one 15.36MHz clock cycle. The clock signal C15_36MHZ drives a counter. The counter counts from 0 to 15 repetitively. The state of the output signal CLK is toggled on each 15^{th} counter state. Figure 30 demonstrates the operation. The frequency of CLK is therefore 1.95μs (65ns times 30) or 510kHz.

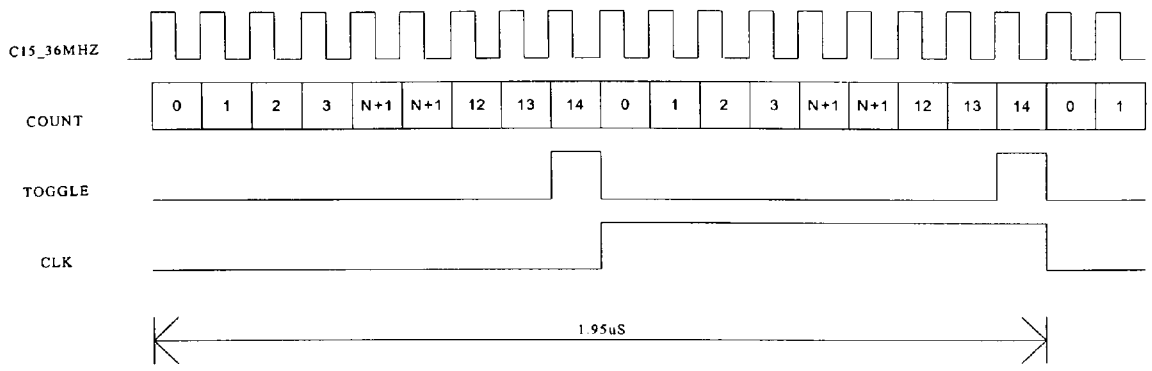


Figure 30 - Timing diagram for the MCLK_DIV block.

The RAM block.

The Random Access memory (RAM) block contains the storage capacity for 1024 bytes. This is broken down as follows. There are 512 samples in one data-acquisition process. Each sample is 16 bits of data. Therefore, the required storage space is, (512 X 16)/8 =1024. The block diagram for the RAM circuit block is shown in appendix a, figure 2. The RAM circuit block is composed of two synchronous 256 X 16 bit primitive RAM cells. These memory cells will be referred to as memory bank A and memory bank B. The signal RAM_SELECT is used to select either bank A or B. When RAM_SELECT is low bank A is selected and when RAM_SELECT is high bank B is selected. Data is written to RAM on the RAM_DIN[15:0] signal bus. Data is read from RAM on the RAM_DOUT[15:0] data bus. A mutliplexer is used to select which memory bank writes to RAM_DOUT[15:0] based on the RAM_SELECT signal state. When RAM_SELECT is low bank A is routed to RAM_DOUT[15:0]. The inputs to a RAM cell are WR_EN, D[15:0], WR_CLK, A[7:0]. WR_EN is the write enable signal and when set high allows write access to the internal memory. WR_CLK is the clock signal running at 512kHz in data-acquisition mode and at 64Hz in data-upload mode. D[15:0] is the input data bus and

data to be written into the RAM is placed here. A[7:0] is the memory address bus and allows access to each of the 16-bit memory locations in memory. The signal D0[15:0] is the output data bus. Data is read from the output data bus when WR_EN is low.

The RAM_ADD block.

The RAM_ADD block implements the RAM address count sequence. A nine bit synchronous counter provides the required 512 binary states, 0 to 511. The counter is enabled by the signal EN_CNT and reset to state 0 by a high on the signal RST_CNT. The signal RAM_ADD[7:0] is the first 8 bits of the binary counter. The signal RAM_SELECT is derived from the the 9^{th} bit of the binary counter and selects RAM bank A or B. RAM_SELECT is set to logic 0 when the count sequence is between 0 and 255 and set to logic high when the count sequence is between 256 and 511. The count sequence is decoded and the signals CNT_512, CNT_256 and ADC_RDY are produced and function as status input signals to the system state machine. The signal CNT_512 is active when the count sequence reads 511. This signal informs the state machine that all 512 RAM locations have been processed. The signal CNT_256 is active when the count sequence reads 255 and facilitates the DAQ card timeout during DAQ data-upload mode. The signal ADC_RDY is active when the counter reads 6. This signal indicates that the ADC data latency timeout during data-acquisition mode is complete.

3.3 The ADS807 circuit block.

The ADS807 block performs the interfacing tasks with the ANALOGUE and DAC_CARD interfaces. The ADS807 contains the state machine controller and hardware architecture for both the data-acquisition mode and the DAQ data-upload mode. The ADS807 block operation was designed to comply with the timing diagram details already presented. Following is a description of the hardware architecture circuit blocks required to facilitate the state-machine operation. The block diagram for the ADS807 block is shown in appendix a, figure 3. The architectural blocks are the 16BIT_BUFF block, the SIGNAL_MUX block, the EDGE_DETECT block, the ADC_ARCH block, and the DAC_CLK_DIV block.

The 16BIT_BUFF Block.

The 16BIT_BUFF block synchronises the incoming data with the clock signal CLK.

EDGE_DETECT Block.

The EDGE_DETECT block implements an edge detector. When a negative to positive transition occurs on the signal START the signal START_EDGE is asserted high for one clock cycle. When a negative to positive transition occurs on the signal DAC_CLK the signal DAC_CLK_EDGE is asserted high for one clock cycle.

DAC_CLK_DIV Block.

The DAC_CLK_DIV block is clocked by the clock signal CLK. A count sequence is advanced from 0 to 7999 when the signal EN_DAC_CLK is high. The timing diagram in figure 31 is not to scale but the relationship between the clock

edges is clearly shown. The count sequence will only advance when EN_DAC_CLK is asserted high. When the count sequence reads 3999 the signal DAC_CLK is set high synchronously with CLK. When the count sequence reads 7999 the signal DAC_CLK is set low synchronously with CLK. The resulting frequency of DAC_CLK is calculated as follows. The frequency of CLK is 512kHz (1.95μs). The frequency of DAC_CLK is the reciprocal of 1.96μs times 8000 and is 64Hz at 50% duty cycle.

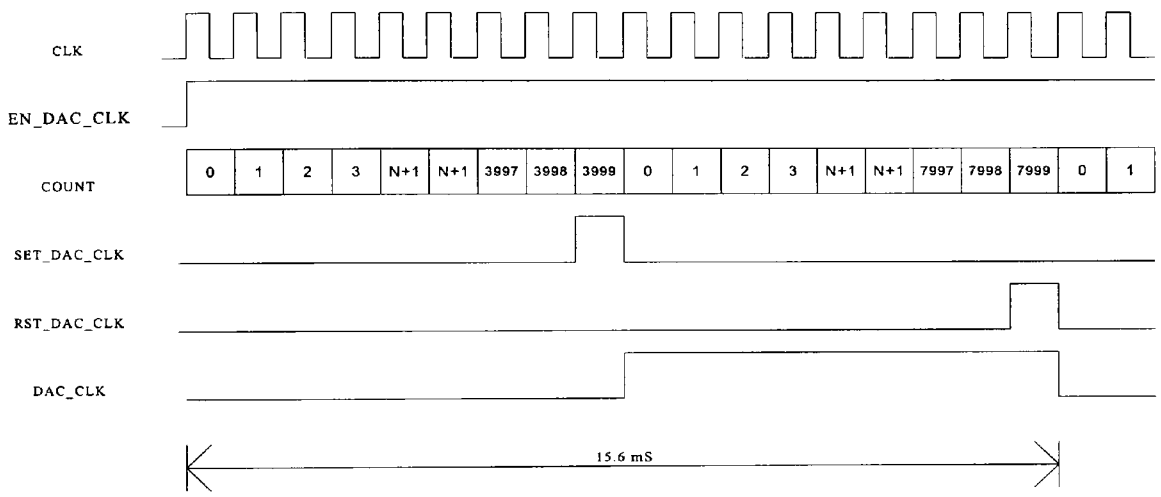


Figure 31 - The DAC_CLK_DIV timing diagram.

The ADC_ARCH block.

The ADC_ARCH block sets the signal ADC_DONE high when SET_DONE is active and resets ADC_DONE low when RST_DONE is high.

The SIGNAL_MUX block.

The SIGNAL_MUX concatenates the 12 bits from the ADC converter with the 4 bit status word into a 16 bit word. The lower 12 bits (RAM_DATA0 to RAM_DATA11) contain the ADC data and the upper 4 bits (RAM_DATA12 to RAM_DATA15) contain the status information. The 4 bit status word can convey 16 different messages although provision is made for only two messages in this application. When the analogue input signal exceeds the ADCs input range the signal OTR is driven high by the ADC and the state-machine asserts the signal OUT_OF_RANGE high. When the signal OUT_OF_RANGE is high the status word RAM_DATA12 through RAM_DATA15 is encoded as 1111. When the OUT_OF_RANGE is low the status word RAM_DATA12 through RAM_DATA15 is encoded as 0000.

The ADC_CON circuit block.

The ADC_CON circuit block implements the algorithmic state machine (ASM) controller for the system. The following is a description of the conventions required to read the ASM chart. A state representation is represented in figure 32. A state is represented by the rectangle and the state name (S1 in this case) appears on the top right hand of each state rectangle. Inputs to the state machine are represented in the diamond. A logic low (0 volts) is called false and a logic high (3.3 volts is regarded as true. Outputs from the state machine may be dependent upon the active state, in this case OUTPUT SIGNAL A and OUTPUT SIGNAL B. Outputs may also be dependent upon the current state and the state of the inputs. For example, OUTPUT SIGNAL C, will go active high when S1 exists and the input signal, INPUT_SIGNAL

is true. If the INPUT SIGNAL, is false while state S1 is active then the state machine will remain in state S1. If INPUT SIGNAL is true while state S1 is active then the state machine will leave state S1. Transitions are made on the positive edge of the master clock signal.

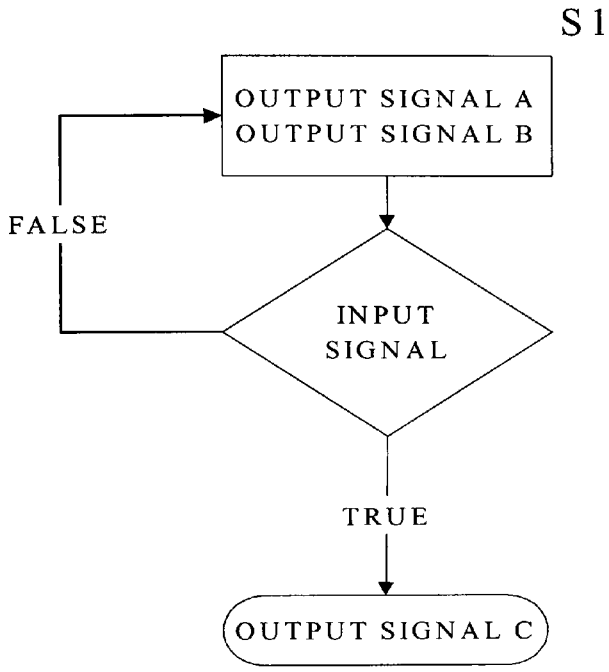


Figure 32 - ASM chart conventions

The ASM chart for the ADC_CON circuit block is shown in figure 33. All inputs are synchronised with the master clock (CLK) and all outputs will occur in synchronism with CLK. The states have already been defined in the timing diagrams of figures 27, 28 and 29. The inputs to the state-machine are defined as follows.

START_EDGE: This signal lasts for a duration of one clock cycle and indicates a negative to positive transition on the signal START. When the state machine is in state S1 START EDGE will cause the state-machine to enter data-acquisition mode.

When the state machine is in state S5 START_EDGE will cause the state-machine to enter DAQ data-upload mode.

DAC_CLK_EDGE: This signal lasts for a duration of one clock cycle. When the state-machine is in state S6 through S10 a pulse on DAC_CLK_EDGE indicates a negative to positive transition on DAC_CLK.

OTR: OTR is a signal from the ADC in the analogue block. This signal indicates the result is in error due to the measured analogue voltage exceeding the input voltage range of the ADC.

CNT_7: CNT_7 is produced by the RAM_ADD block and lasts for a duration of one clock cycle. When active, this indicates that the ADC latency time-out has passed.

CNT_256: CNT_256 is produced from the RAM_ADD block and lasts for a duration of one clock cycle. CNT_256 indicates that the DAQ time-out has passed and goes active when the RAM_ADD count sequence has cycled through 256 clock cycles. When the state-machine is in state S7 and CNT_256 is true a transition is made to state S8.

CNT_512: CNT_512 is produced from the RAM_ADD block and lasts for a duration of one clock cycle. CNT_512 indicates that all 512 RAM locations have been processed. When the state-machine is in state S9 and CNT_256 is true a transition is made to state S10.

CLK: This is the clock signal and runs at a frequency of 512kHz and 50% duty cycle.

The outputs from the state-machine are defined as follows.

SET_DONE: This sets the signal ADC_DONE to logic high.

RST_DONE: This resets the signal ADC_DONE to logic low.

OUT_OF_RANGE: This signal indicates that the current ADC result is outside the range of the ADC input voltage range. OUT_OF_RANGE signals to the SIGNAL_MUX block that the status word associated with the current result should be set to all logic ones.

EN_CNT: This enables the RAM_ADD blocks count sequence to advance to the next state.

RAM_WE: When this is set to logic high the RAM is enabled for a write operation.

OE: This is a signal to the ADC in the analogue block and a logic low enables an ADC read operation.

EN_ADC_CLK: This enables the sampling clock to the ADC in the analogue block.

DAC_CLK_ON: This enables the DAC_CLK_DIV block and the 512KHz clock signal is divided down to 64Hz.

EN_DAC_CLK: This enables the DAC_CLK to the DAQ card.

The state encoding for the state-machine is implemented using a one-hot encoding technique. In a one-hot encoded state machine a flip-flop is employed for each state. In the current implementation eleven states exist. Only one flip-flop can be active at any one time and at power-up or when a reset occurs the system defaults to state S1. During the state machine operation if more that one flop-flop becomes hot then the state machine will produce erroneous results. This is overcome by adding extra circuitry to monitor for this situation and force a reset if identified. The following ASM reflects the state machine operation and the VHDL code can be implemented directly from it

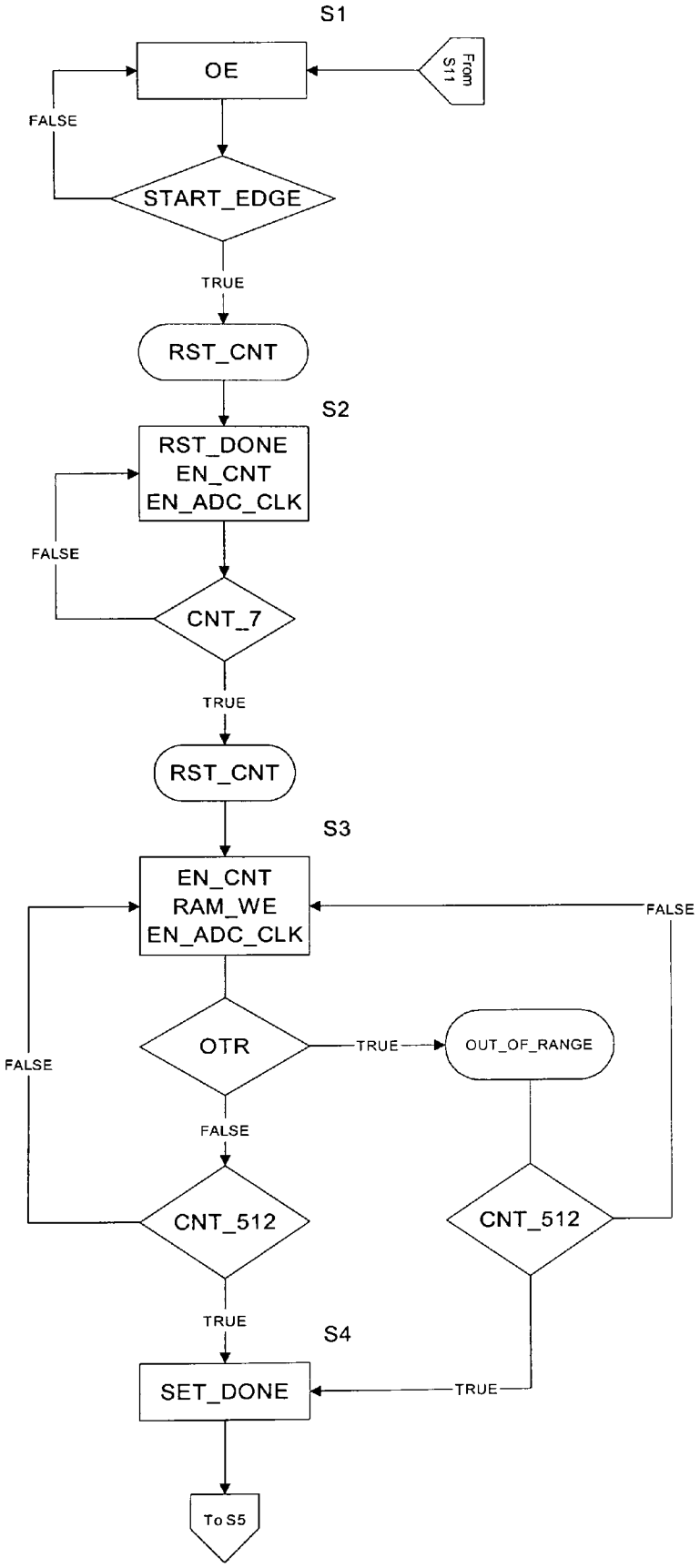


Figure 33 – The ADC_CON ASM chart – Part A.

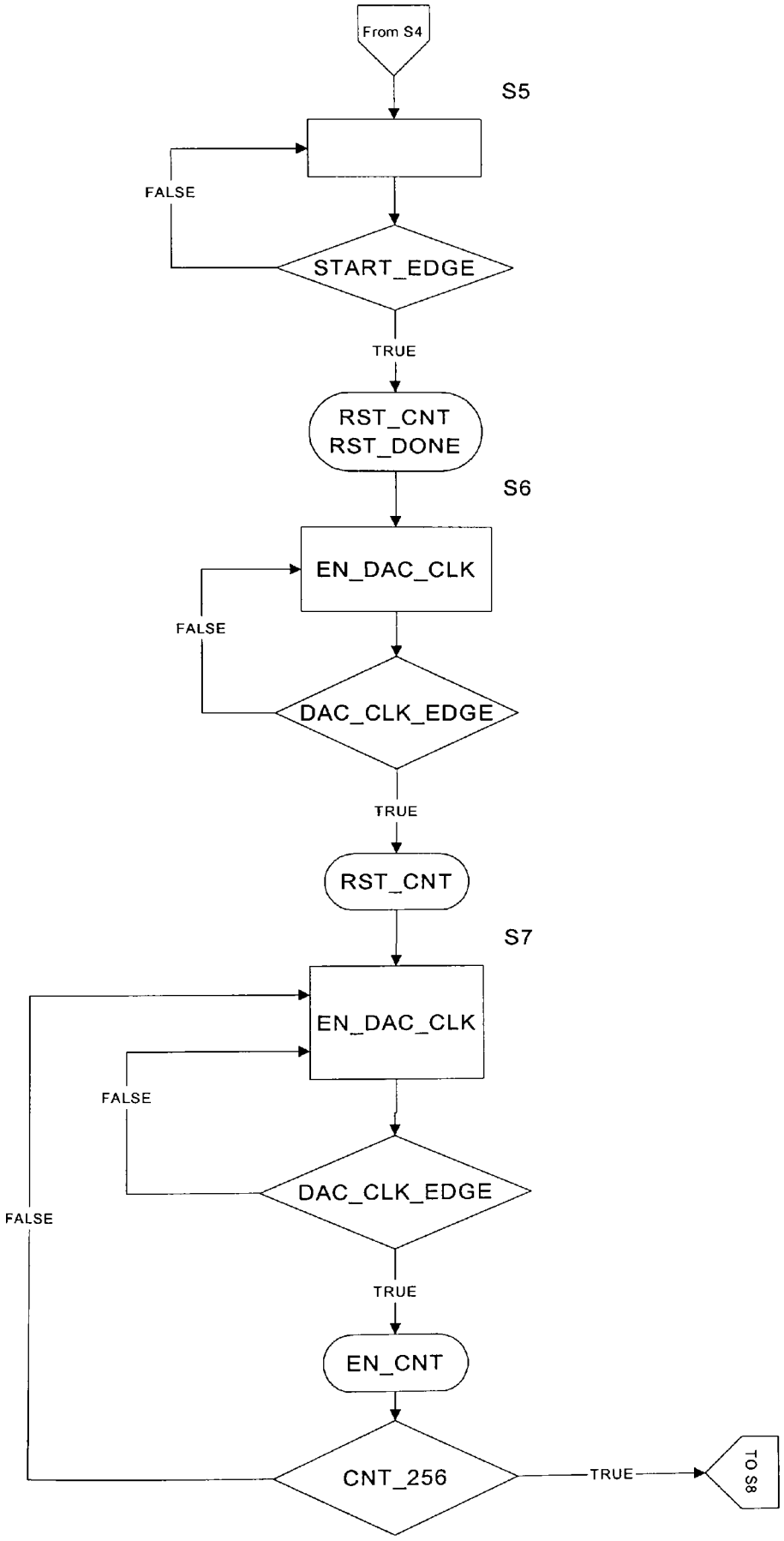


Figure 34 – The ADC_CON ASM chart – Part B.

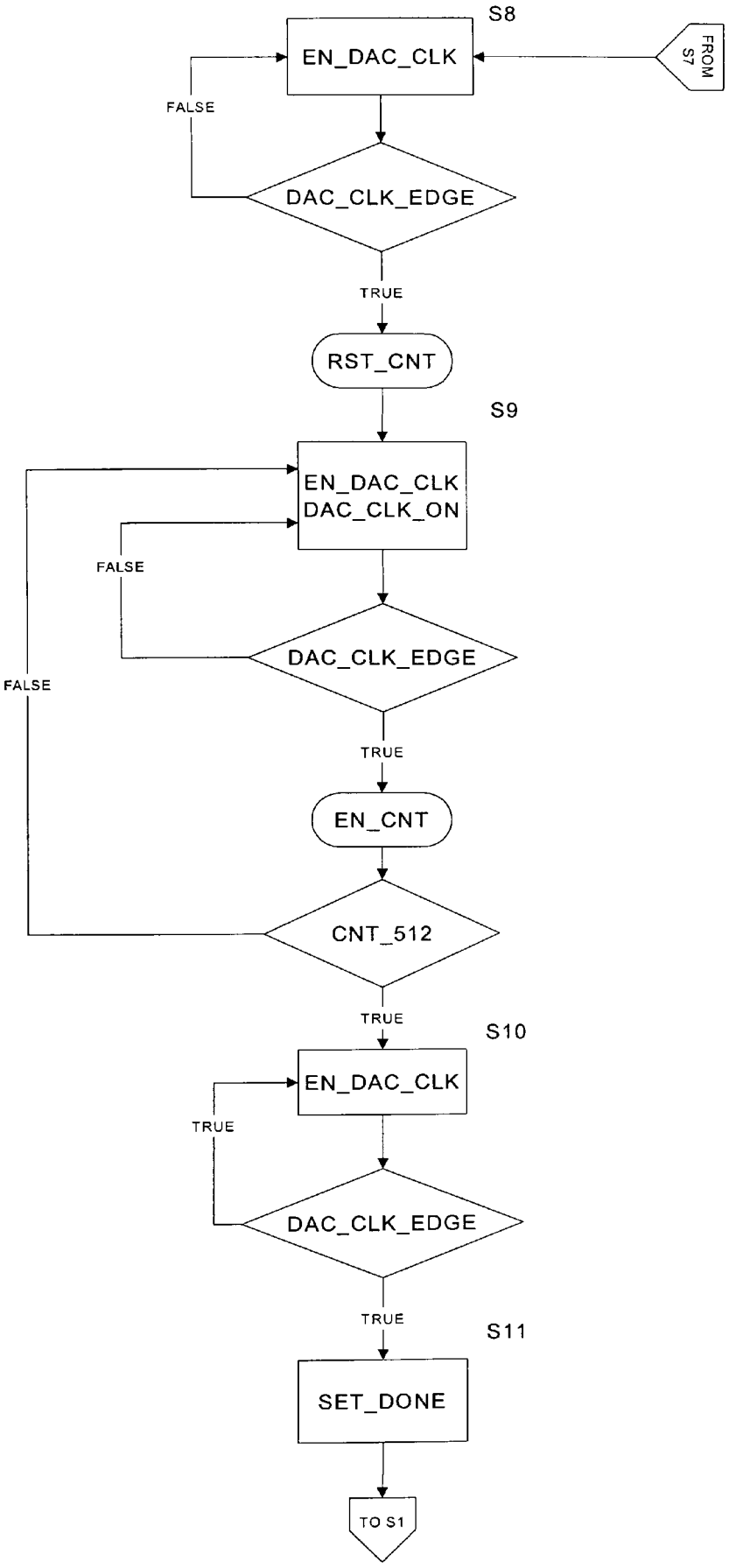


Figure 35 – The ADC_CON ASM chart – Part C.

FPGA implementation details.

The FPGA firmware design targeted a Xilinx 5010XL/Spartan device. The device utilization summary is as follows.

Number of External IOBs 36 out of 61 59%

Number of Global Buffer IOBs: 1 out of 8 12%

Number of CLBs: 362 out of 400 90%

Total Latches: 0 out of 800 0%

Total CLB Flops: 54 out of 800 6%

4 input LUTs: 709 out of 800 88%

3 input LUTs: 282 out of 400 70%

Number of BUFGLSs: 2 out of 8 25%

Number of TBUFs: 256 out of 880 29%.

3.4 The analogue Block.

The analogue block measures the output signal from the DUT. A block diagram of the system is shown in figure 36. The signal conditioning block attenuates the DUTs signal so that it fits within the input voltage range of the ADC. A differential amplifier converts the differential signal from the DUT to a single ended signal while removing common mode noise. The ADC converts the single-ended 2B1Q signal to a to a digital signal with a resolution of 12 bits. A transformer is required to isolate the electronics in the receiver from harmful voltages that a faulty DUT may output during a PSD measurement.

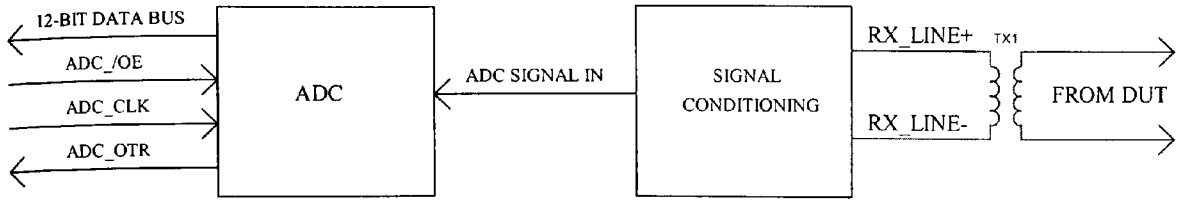


Figure 36 - Block diagram of the receiver analogue electronics.

The Analogue to digital converter.

The selection criteria adopted for the ADC considered sampling frequency, ADC accuracy and resolution, analogue input range, and the digital interface method. Once the ADC was selected the signal conditioning amplifier was chosen to match.

The sampling frequency.

The 2B1Q reference signal transmitter is configured to transmit at 160kbps (80kHz). The reference signal transmitter ensures the transmitted signal power is contained below 240kHz on the frequency range. This is due to the sharp filtering of the output 2B1Q signal in the AFE1224 chip. To test the system it was decided to monitor the spectrum up to 256kHz. In order to calculate the PSD with a resolution of 1kHz over this range a sampling frequency of 512KHz is required to measure PSD for basic rate ISDN applications. The reasoning to support this statement was presented in the portfolio overview. The ADC sampling frequency was also chosen to ensure that the PSD of DSL systems other than basic rate ISDN could be measured. Currently the maximum bit rate provided by ISDN systems employing the 2B1Q line code is 2.048Mbps. A system transmitting at 2.048Mbps operates at a frequency of 1.024MHz. The power spectrum would contain most of the signal power below

1.024MHz. Providing a sampling frequency of 10MHz would allow calculation of the PSD over the frequency range up to 5MHz. Choosing an ADC with a sampling frequency range of 10kHz to 20MHz could provide for the wide-ranging operating frequencies of currently deployed systems.

ADC Resolution.

In order to obtain the required resolution for the measurement system, the 2B1Q pulse-mask defined in the T1.601 standard is observed in order to obtain the worst-case voltage swing. The maximum voltage swing allowable is +/-2.625V. This gives 5.25V peak to peak. The lowest voltage is +/- 0.00833 (16.6mV peak to peak). From this the required system accuracy is 16.6mV divided by 5.25V to give 3.16mV or 0.316%. A factor of 10 has been added here to act as an Engineer's tolerance. 0.316% divided by 10 gives 0.0316%. The ADC resolution was chosen based on a 0.0316% accuracy value. A 12-bit ADC provides an accuracy of 0.0244%. This is calculated as follows.

$$\text{ADC accuracy} = \frac{1}{2^N} \cdot 100$$

Where N = 12 bits.

12 bits is thus adopted in this application. The ADS807 ADC produced by Texas Instruments was chosen to satisfy the resolution and sampling frequency requirements. The ADS807 data sheet is referenced in [7]. The following analysis looks at the errors inherent in the ADC and its choice is validated based on an error budget analysis.

The ADS807 analogue to digital converter.

The ADS807 can operate at a sampling-rate of 53MHz. A differential analogue stage is provided but this facility was not used and the analogue input stage was configured as single ended. The ADS807 has a resolution of 12 bits. The Effective Number of Bits (ENOB) is dependant on the signal to noise ratio plus distortion and is defined as,

$$ENOB = \frac{SINAD(dB) - 1.76}{6.02}$$

SINAD is a measure of Signal to Noise Ratio (SNR) and distortion (D) and offers a more realistic signal to noise ratio. SINAD can be expressed as follows.

$$SINAD = SNR + D = \frac{1}{THD + N}$$

THD is total harmonic distortion and is a ratio of energy in the fundamental signal to the energy in the harmonics. N is the ADC resolution. The SINAD value for the ADS807 can be derived from the following graph presented in the datasheet and is reproduced here in figure 37.

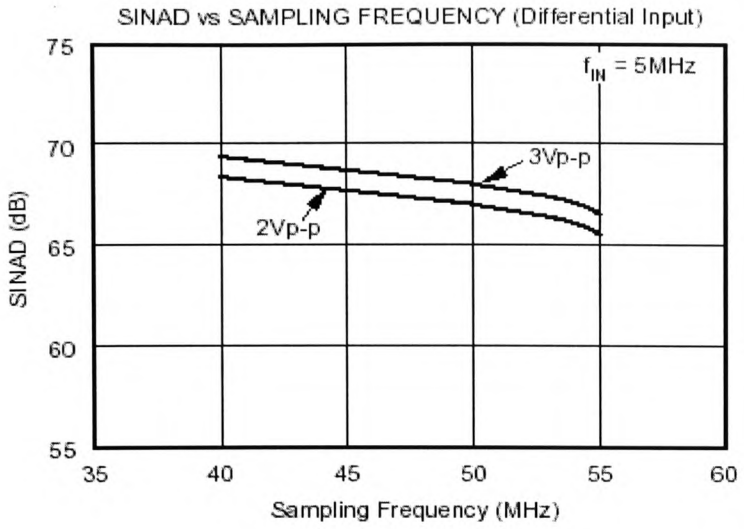


Figure 37 - Graph of SINAD against sampling frequency.

The SINAD value stated in the ADS807 is for a differential input however there is no information for the single ended case where the input frequency range is 80kHz to 1.024MHz. If the curve labeled 2Vp-p is adjusted to reflect the single ended case and extended back beyond the 35MHz sampling-rate point then a SINAD value of 68dB for lower sampling rates may be possible. This would produce an ENOB value of 11 bits and an accuracy of 0.049%. The datasheet states that the SINAD value is 64dB for a sampling frequency of 40MHz and an input frequency of between 1MHz and 10MHz measured using a single-ended input. This would yield an ENOB of 10 bits. The accuracy would then be reduced from 0.0244% to 0.098%. From the above deductions the actual accuracy is expected to range from 0.049% to 0.098%. With the required measurement accuracy of 0.316%, the worst-case accuracy of 0.098% would still allow for a 0.218% (0.316% - 0.098%) error budget.

The ENOB value offers an approximate accuracy value for the ADC and in reality may vary depending on sampling rate and ADC input frequency. An analysis of the dc specifications for the ADC component involves consideration of Differential

Non-Linearity Error (DNL), Integral Non-Linearity Error (INL), the reference voltage error and the drift of the voltage reference with temperature. The ADS807 data sheet specifies DNL as being 0.5LSB (0.0122%) and 1LSB (0.0244%). The percentage error is calculated as %error = LSB error divided by 2^{12} multiplied by 100. INL is specified as being between 2LSBs (0.0488%) and 4LSBs (0.0976%). The references can have a gain and zero error of 0.1% at 25°C (according to Texas instruments applications support). The zero error drift with temperature is stated as 16ppm/°C or (16ppm/°C x 30^0 x 10^{-6}) 0.048%. The gain error drift is stated as being 66ppm/°C or 0.198%. Calculating the route of the sum of the squares gives an indication of the expected achievable accuracy.

$$\sqrt{(0.0244)^2 + (0.0976)^2 + (0.01)^2 + (0.01)^2 + (0.0488)^2 + (0.0976)^2} = 0.2\%$$

This gives an error budget of 0.3% - 0.2% = 0.1%. In reality the real error contribution should be much less that those specified above.

The ADC interface.

Measurement data is output from the ADC via a 12-bit parallel data bus. An external clock is provided (in this case by the FPGA) and the clock frequency defines the sampling rate. The data-sheet for the ADS807 specifies the clock duty cycle as 50%. The ADC analogue input signal is sampled on the rising edge of the clock and is valid on the negative edge of the clock. The ADC timing diagram is shown in figure 38. Data Out on the timing diagram represents the changing 12-bit data bus. The

output code format is straight offset binary code. The code output with respect to the analogue full-scale (FS) input is as follows.

Analogue input range	Digital code
+ Full scale input	111111111111
Half scale input	100000000000
- Full scale input	000000000000

The signal /OE is the output enable for the data bus and when high places the data bus in a high impedance state.

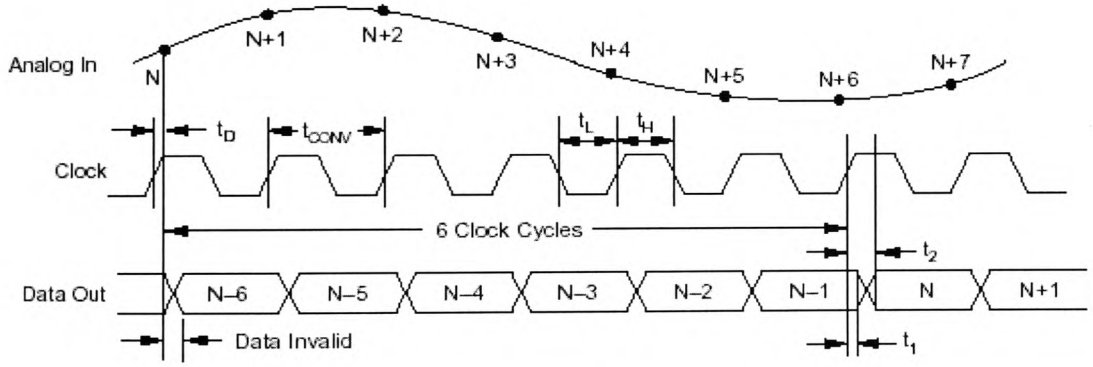


Figure 38 - ADC timing diagram.

ADC implementation details.

The ADC input voltage range is set as 2V by pulling the FS_{SEL} pin to logic low. Pin 18 is set low and the internal references are selected. An external reference has also been provided on the Printed Circuit Board (PCB) and can be selected by setting FS_{SEL} high. The system employs a differential front-end amplifier and is thus

configured to make a single ended measurement. The ADC wiring configuration is shown in figure 39. The negative input (pin 24) is pulled to the common mode voltage point (2.5V). The input signal is coupled to the common mode voltage point via C31 and applied to the positive voltage input on the ADC.

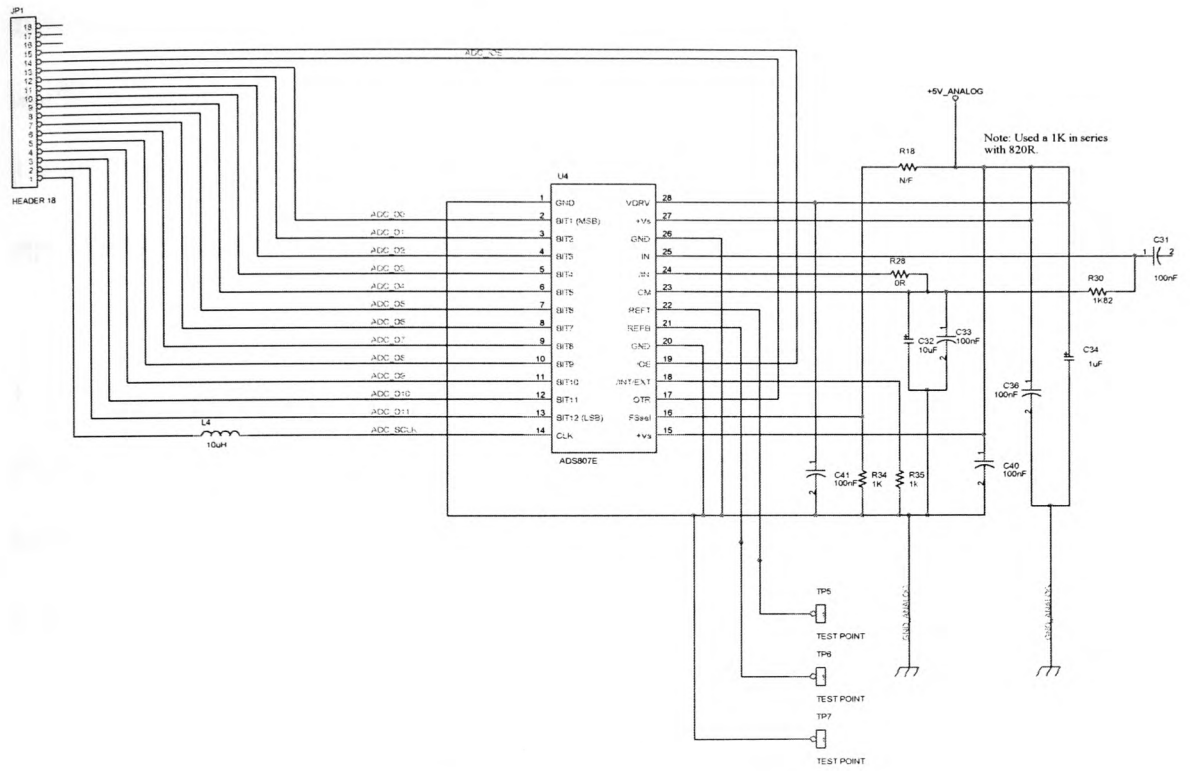


Figure 39 - ADS807 wiring configuration.

Analogue front-end signal conditioning.

The analogue front-end signal conditioning is composed of a transformer, an attenuator and a differential amplifier circuit. This equates to three stages that the DUT input signal must pass before being applied to the ADC. The ADC input range is 2V and the maximum voltage swing from the DUT can be 3V peak to peak. The analogue front end is therefore required to attenuate the DUTs voltage before an ADC

measurement can be achieved. Figure 40 presents the voltages at each stage in the front-end signal conditioning system. The voltage across the 135ohm resistor is that transmitted by the DUT. This is a differential signal and the two signal components are presented separately in figure 40. Each signal component has the same amplitude as shown but one signal component is 180° out of phase with respect to the other. If a difference amplifier were connected across the 135ohm resistor, a single ended signal would be produced. However, the output voltage would swing between +3 volts and – 3 volts yielding a peak-to-peak voltage of 6 volts. This is far in excess of the ADC input voltage range and attenuation of the signal is required.

The transformer will attenuate the signal by 1.6 due to the transformer ratio of 1:1.6. Between stage A and stage B a potential divider set to attenuate by a factor of three is employed. The output of stage B is applied to the difference amplifier and the full-scale voltage swing applied to the ADC is 1.24V peak-to-peak. This provides 0.38 volts of headroom above each peak (2V – 1.24V = 0.76).

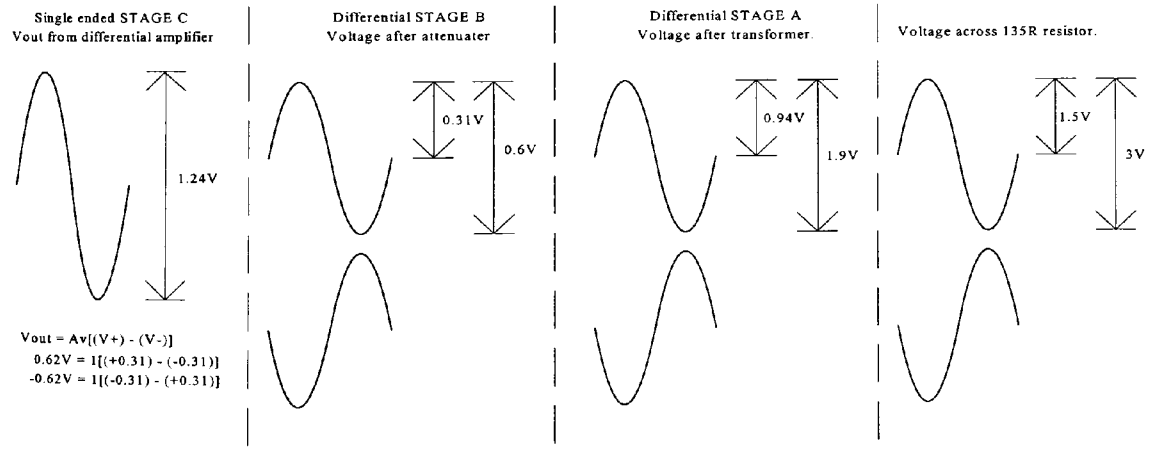


Figure 40 - The voltages at each stage in the signal conditioning system.

In order to test the signal-conditioning circuit, the 2B1Q reference signal transmitter was configured to generate a test signal. The test signal consists of repetitive –3-1+1+3+1-1-3 2B1Q voltage levels. This will produce a signal with maximum voltage swing and allow triggering with an oscilloscope. The test signal is applied to the receiver and the signal is measured with an oscilloscope at the output of each stage in the signal conditioning circuit. This is a validation exercise and the measured results are compared with the expected voltage values.

The transformer coupling stage.

The transformer coupling stage employs an APC 42901 ISDN transformer. The turns-ratio is 1:1:6. Figure 41 shows the wiring diagram. The differential signal from the DUT is applied across pins 1 and 3 of CON3. The transformer reduces the signal by a factor of 1.6 and resistors R26 and R27 reference the signal to ground. The capacitors C27, C29, and C30 are not fitted in this case. Provision for their placement was made at the PCB layout stage so that capacitive coupling could be achieved if a transformer is not fitted. Figure 42 shows the differential test signal from the DUT as measured across a 135ohm resistor. Figure 43 shows the differential test signal after the transformer stage.

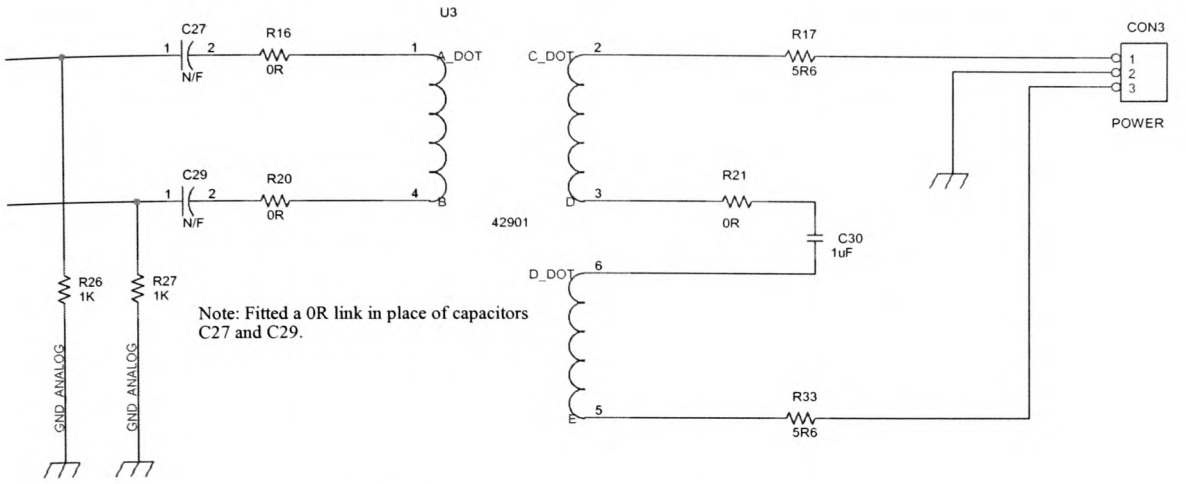


Figure 41 - Transformer coupling stage.

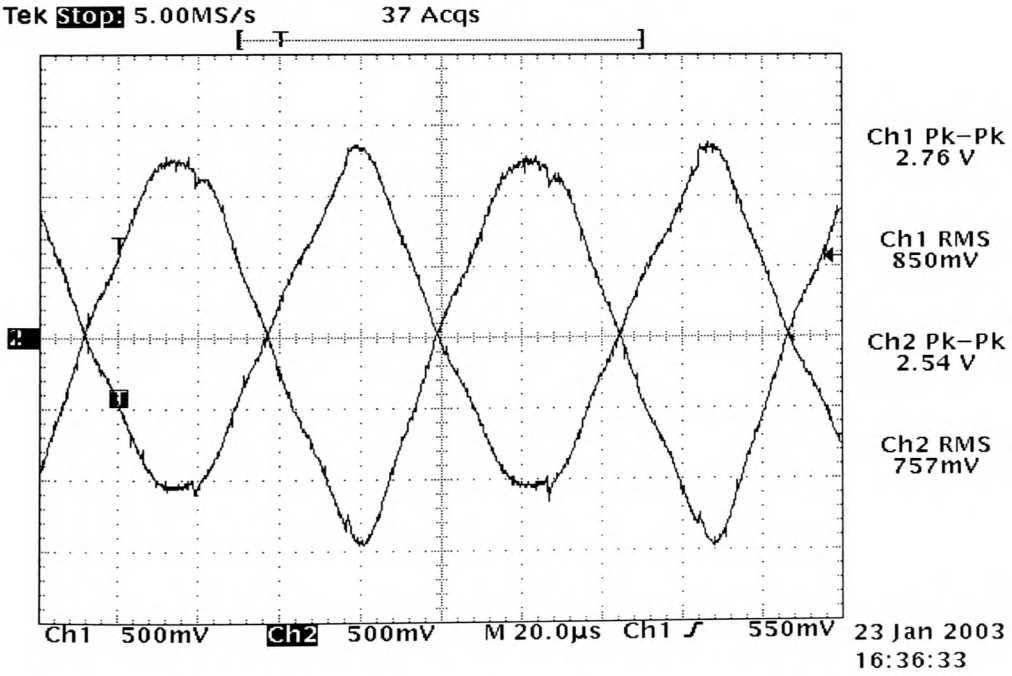


Figure 42 - Test signal from DUT across 135 ohms.

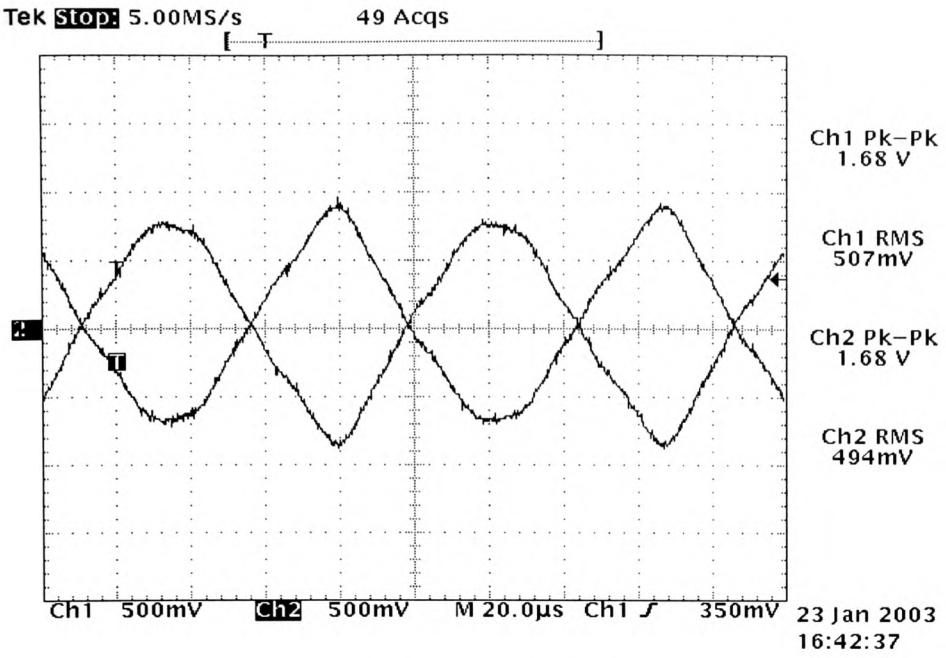


Figure 43 - Test signal at transformer output.

The attenuator stage.

The attenuator consists of a potential divider and buffer amplifier. The potential divider attenuates both signal components by 0.3125 as shown in figure 44. Figure 45 shows the waveforms at the attenuator stage output and an expected 600mV peak-to-peak is measured. The differential signal enters the attenuator stage via resistors R15 and R19. The OPA2682 is employed as a buffer amplifier and is configured to have unity gain [8]. The slew rate indicates the fastest signal that can be buffered by the amplifier before distortion of the amplifiers output signal occurs. In this case distortion would increase the total harmonic distortion and ultimately decrease the ENOB value of the system. From figure 40 the worst case voltage swing from the DUT could be 3 volts in 12.6µs. The potential divider reduces this to 0.625mV in 12.6µs. An Engineer's tolerance was added to this figure and the worst case voltage swing value used in calculating the required slew rate was (0.625mV

times 10) 6V/12.6µs. This equates to a required slew rate of 476mV/µs. The OPA2682 has a slew rate of 570V/µs.

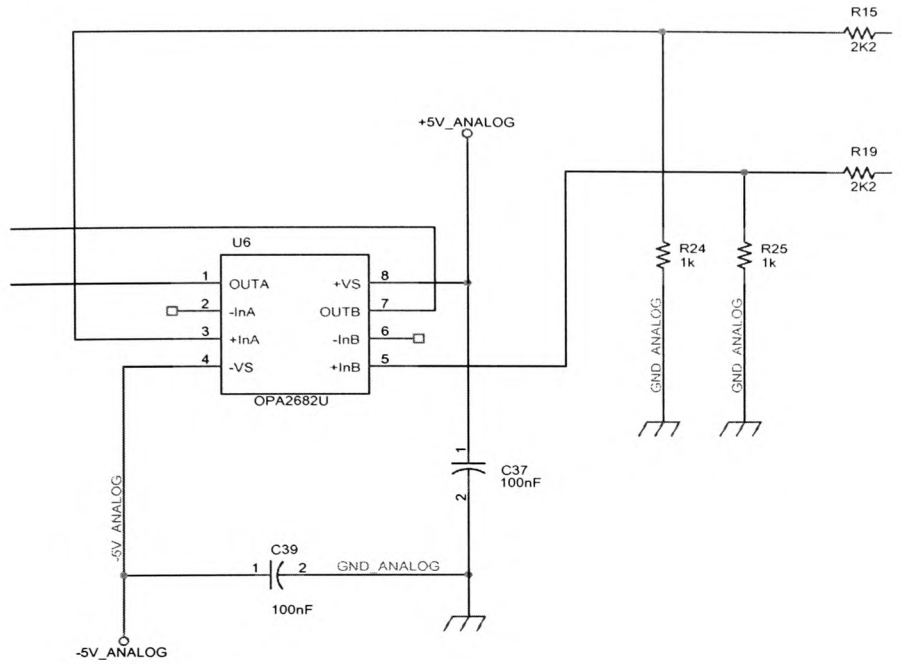


Figure 44 - Attenuator stage.

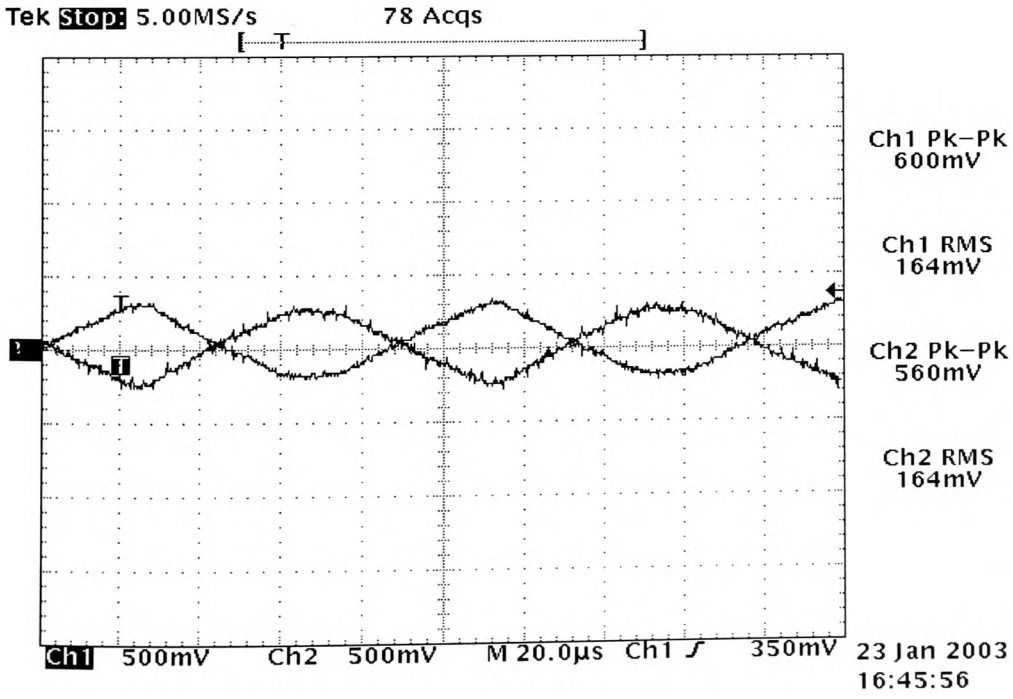


Figure 45 - The attenuated and buffered test signal.

The differential amplifier stage.

An OPA620 is configured as a differential amplifier with a gain of 1 as in figure 46. The amplifier output derived from the differential amplifier formula as presented by Jacob [9].

$$Vout = Av((V+) - (V-))$$

In this case the amplification factor Av is set to 1. The positive terminal of the amplifier, V+ can swing between +250mV peak and –250mV peak. The negative terminal has the same signal amplitude but is 180° out of phase with respect to the positive terminal. The amplifier equations are evaluated for the full voltage swing as follows,

$$+500mV = (+250mV) - (-250mV)$$

$$-500mV = (-250mV) - (+250mV)$$

This yields a full peak-to-peak voltage swing of 1V. The oscilloscope plot in figure 47 shows a peak-to-peak voltage swing of 1.02V. In the case of a pseudo random 2B1Q signal it was observed that measured voltages vary approximately by 250mV above 1.02V. The 2B1Q pseudo-random waveform transmitted from the DUT is shown in figure 48. Figures 49, 50 and 51 show the 2B1Q pseudo-random waveforms at the output of each stage in the design.

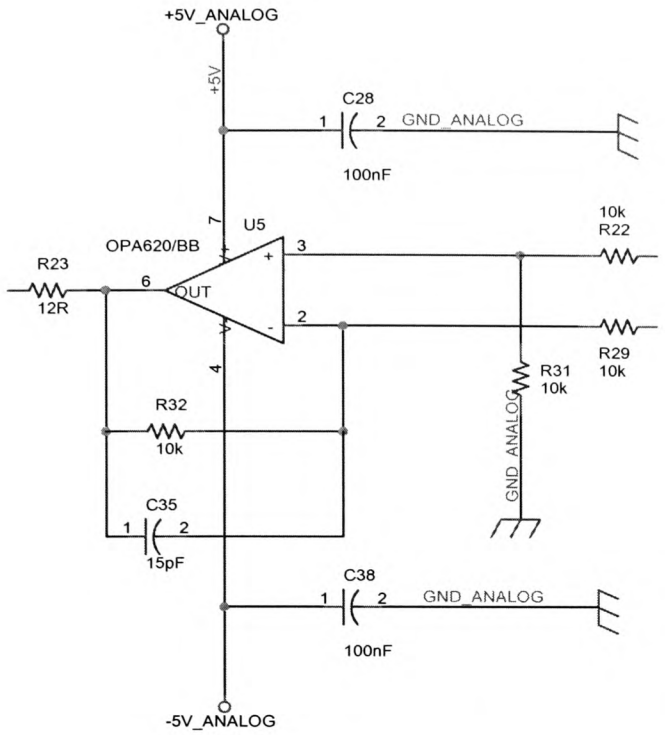


Figure 46 - The differential amplifier stage.

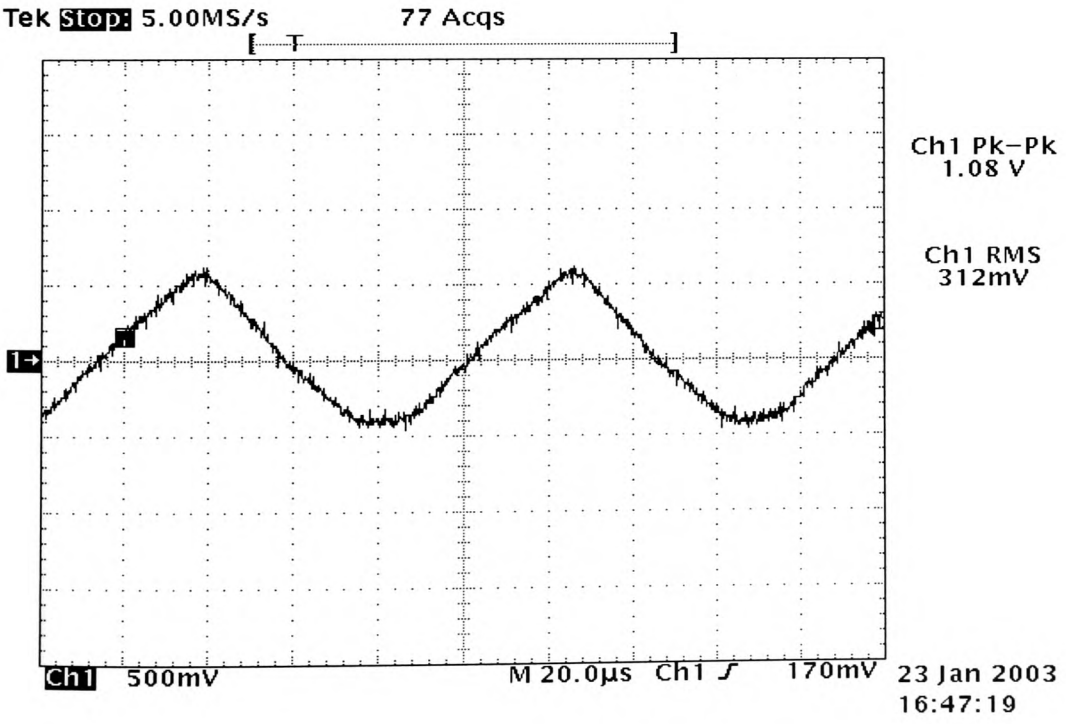


Figure 47 - Differential amplifier output.

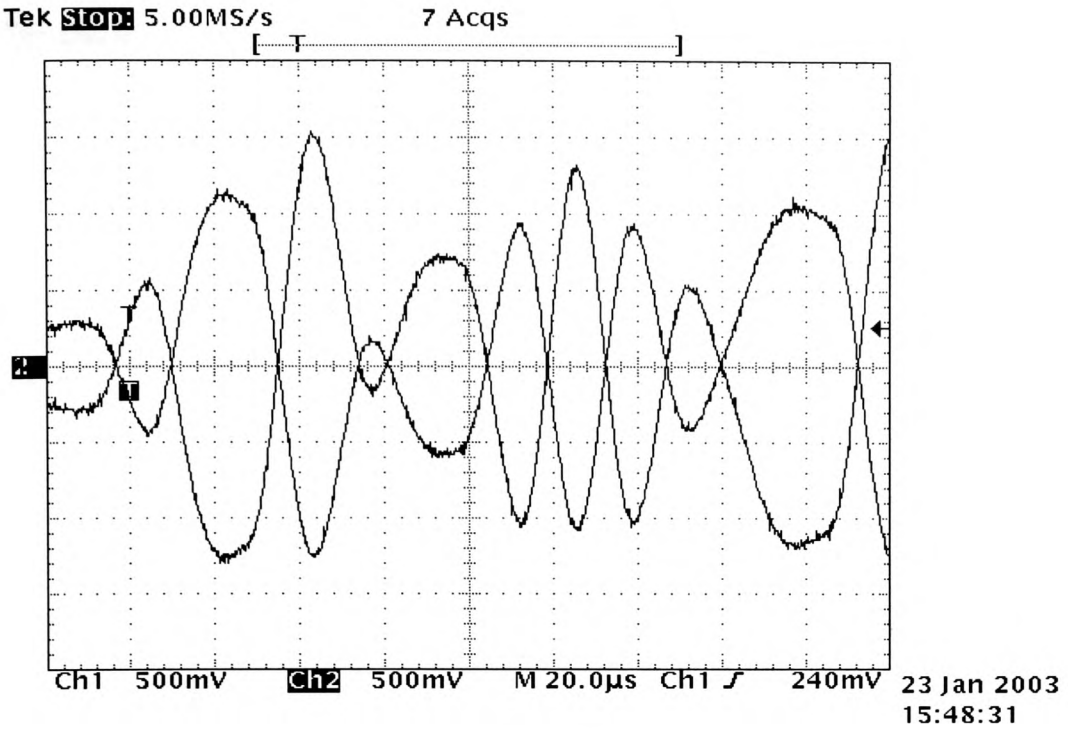


Figure 48 - Differential pseudo-random 2B1Q signal across 135 ohms.

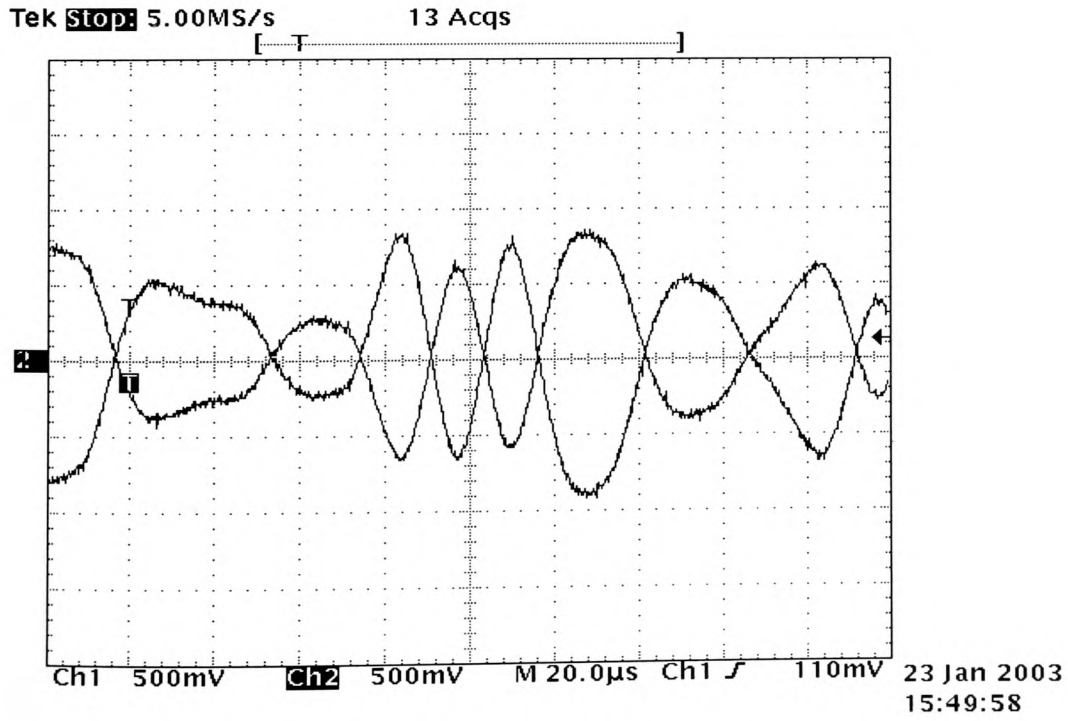


Figure 49 – Differential 2B1Q signal measured at output of transformer.

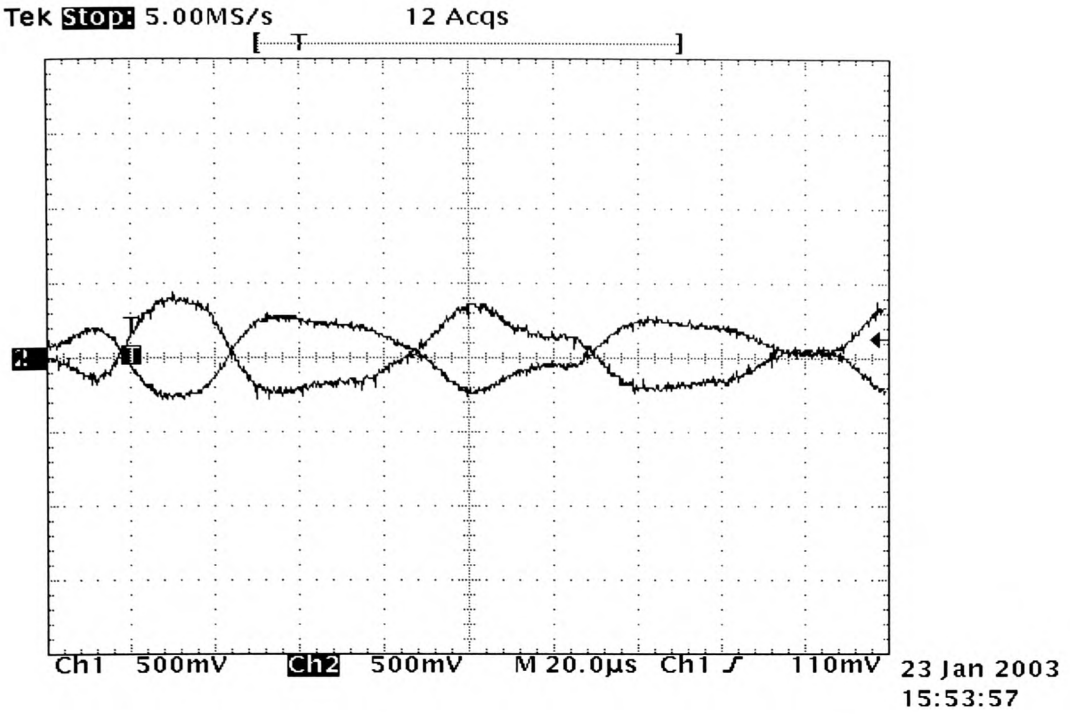


Figure 50 - Differential 2B1Q signal at output of attenuator stage.

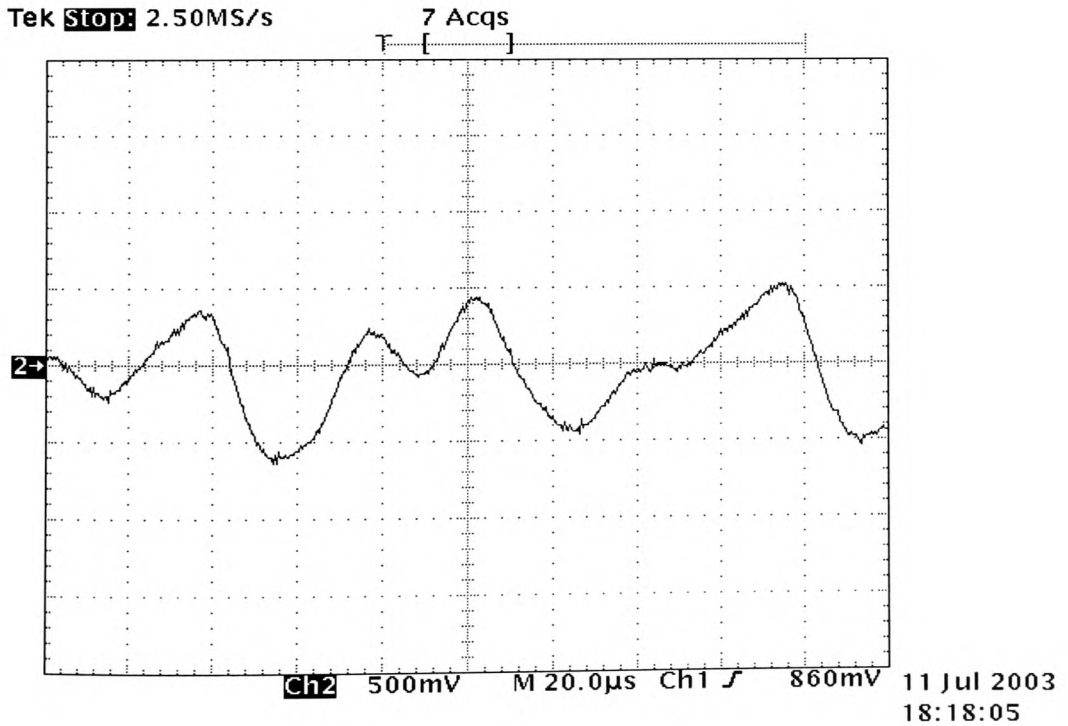


Figure 51 - The OPA620 differential amplifier output.

Chapter 4

4 The PSD measurement software.

The PSD software measurement programme controls the hardware measurement system and performs the PSD measurement on the sampled data sequence. The measurement software initiates the hardware data-acquisition process and retrieves the results. The obtained results are stored as result files on the computers hard disk for analysis. The transmitted voltage waveform from the DUT is sampled a number of times and the spectrum of each measured data sequence is averaged to produce the result. The analysis task estimates the power spectral density from the data files. The user is presented with a graphical display of the measured PSD spectrum along with the associated PSD mask. It is possible to assess graphically if the DUT is transmitting at correct power levels by inspection of the measured spectrum. A pass or fail bi-color indicator is also included on the graphical display and green indicates a pass condition while red signals a failure.

4.1 The software implementation.

The software functionality was implemented using the Labview graphical programming language from National Instruments. Labview is an interpreted language and runs on Microsoft Windows. The term Virtual Instrument (V.I.) is used in reference to software routines. A software routine may have a sub-routine and this is termed a sub-V.I. The software language is event driven and each V.I. is composed of a graphical panel and an associated diagram. The panel is the graphical user

interface and is linked to the diagram via terminals. The panel may contain controls such as buttons that allow the user to generate events and graphs to present data. The diagram contains a schematic that represents the operation of the software. The schematic is composed of sub-V.Is and graphical functions that may be pre-defined or user-defined. The software automatically interfaces with the NI-DAC driver software. The software interfaces to the hardware measurement system via a data acquisition card (National instruments DAQcard 1200). The digital ports of the card are used to generate the control signals for hardware interfacing and for reading the 512 samples of 16-bit data from the measurement hardware.

The software can be partitioned into two separate tasks called Acquire N Points and Periodogram. The first task, Acquire N Points, interfaces to the hardware measurement system and initiates a data-acquisition. Once the data-acquisition process is complete the software retrieves 512 samples of measurement data from the hardware and writes the data to a result file. Each time the task is repeated a new result file containing 512 samples is created and is written to the hard disk. The second task, Periodogram, calculates the power spectrum density from the data stored in each result file. The separate PSD results are averaged to achieve a final result. The resulting PSD is presented to the user in graphical form and a comparison with a PSD mask is made to produce a pass or fail indication.

4.2 The Acquire-N-points Virtual Instrument.

The functionality of the Acquire-N-points V.I. can be represented by the software flow diagram in figure 52. Each rectangular box represents a process within

the task. The VI executes when the user presses the START button on the front panel. The following is an explanation of each process.

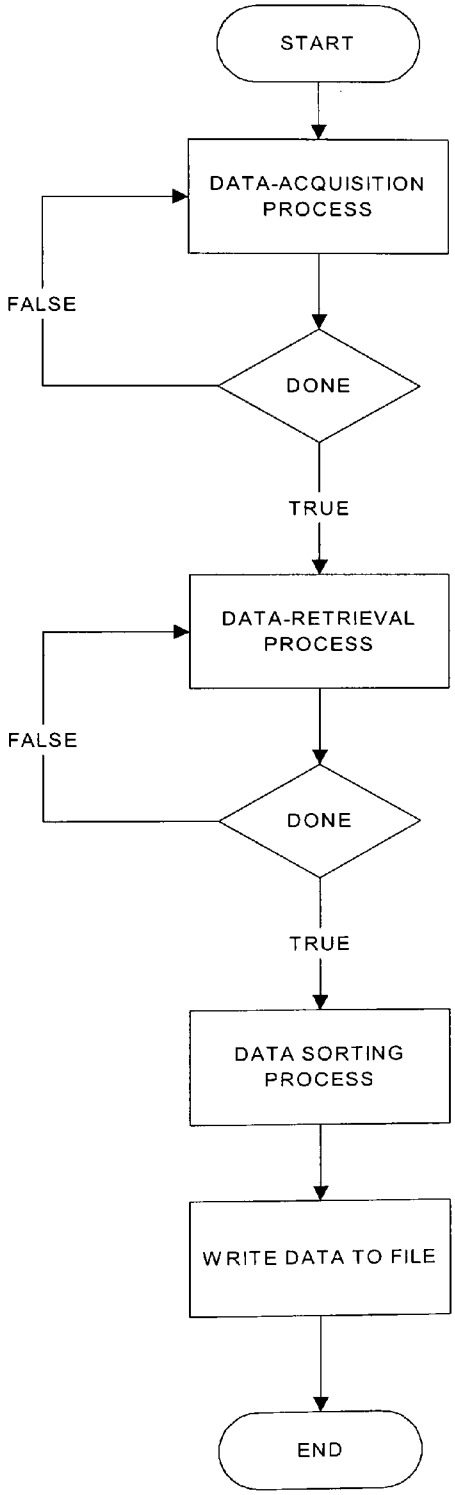


Figure 52 - The Acquire N points V.I. flow chart

The data acquisition process.

The data acquisition process is performed by the ADC sub-VI and the flow chart showing its operation is outlined in figure 53. The sub-VI runs when the signal START ADC goes high. The DAQ cards digital ports are initially cleared and a time-out of 100ms allows NI-DAQ to clear the card before the programme progresses. Digital I/O line 6 on port 2 is set high by the write-one-point-to-digital-line VI. This line corresponds to the signal ADC_START described in chapter 3. Another 100ms time-out passes before the write-one-point-to-digital-line VI sets I/O line 6 on port 2 back to a logic low. At this stage the positive going edge of line 6 will have started the data-acquisition process in the measurement hardware. When the data-acquisition process is complete the DONE signal from the FPGA is sent to the software. The Read-one-point-from-digital-line VI continually reads digital I/O line 7 on port 2. When DONE is set high the data-acquisition process clears the digital ports on the card and executes the data-retrieval process.

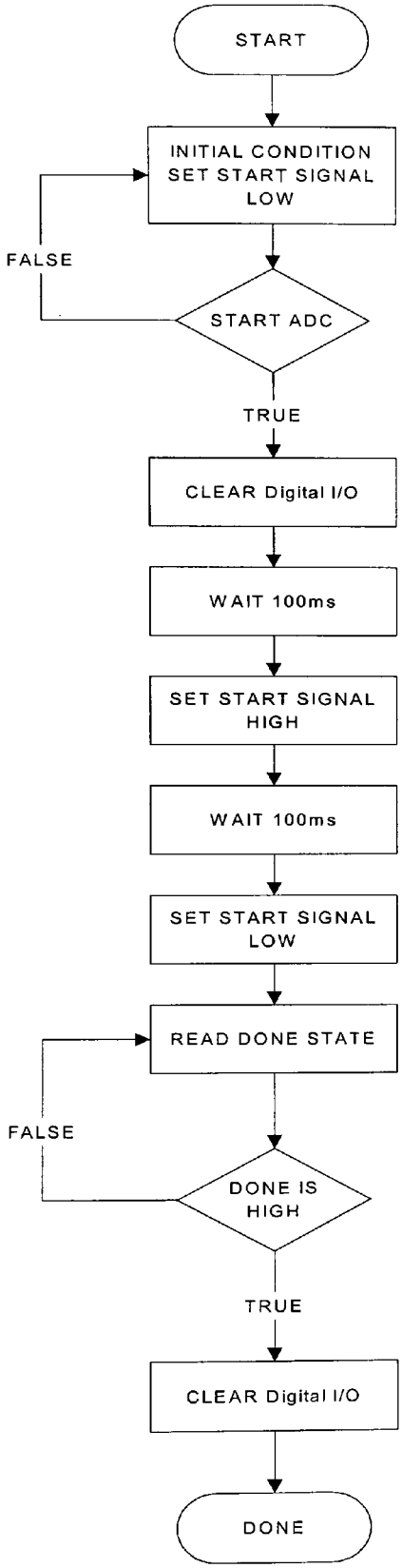


Figure 53 - The flowchart for the ADC V.I.

The data-retrieval process.

The data retrieval is performed by the ACQUIRE_DATA sub-VI. The ACQUIRE_DATA sub-VI drives the digital I/O line 6 on port 2 high and then calls the Dig-buf-handshake-ln(8255) VI. The Dig-buf-handshake-ln(8255) VI can be down-loaded from the National instruments web-site. The VI is configured to concatenate the 8-bit digital I/O ports 1 and 2 as a single 16-bit input port. The VI is set to read 512 samples into the cards on-board buffer every time the sampling clock drops low. This all happens under hardware control and when the process is finished 512 samples are retrieved from the DAQ cards buffer by the Dig-buf-handshake-ln(8255) VI and written to an array for processing. The sampling clock is produced by the FPGA and sent to digital I/O line STBb on port 2 when in data-upload mode. The positive going edge of START_ADC (digital I/O line 6 on port 2) causes the FPGA to enter the data-upload mode.

The data is written to an array during the Data Retrieval process. The data stored in the array is interleaved between port 1 and 2 as shown in figure 54. The Data Sorting process removes all the samples associated with port 1 and places them in a separate array. The same is done for the samples from port 2. The separate arrays are converted from integer to Boolean types and then concatenated as one array of 512 elements consisting of 16-bit words. The Boolean array is then converted into an array of long integers ready for the Write To File operation.

Port1 Sample 1
Port2 Sample 1
Port1 Sample 2
Port2 Sample 2
Port1 Sample N
Port1 Sample N
Port1 Sample 512
Port2 Sample 512

Figure 54 - The array format returned by the data-retrieval process.

The Write data to file process.

The write data to file process employs the Write-to-spreadsheet-string VI to convert the array of long integers to a spreadsheet text string whereby each element in the string is separated by a comma. This text string is written to a text file. The number of the result file is contained in the file name. The result file name is prbs512KHz_N where N is the file number. The first 512 data points are stored in prbs512KHz_1 the second are stored in prbs512KHz_2 and so on.

4.3 The Periodogram Virtual Instrument.

The flowchart outlining the operation of the Periodogram V.I is presented in figure 55. The following is a description of each process. The read data points from file into array process.

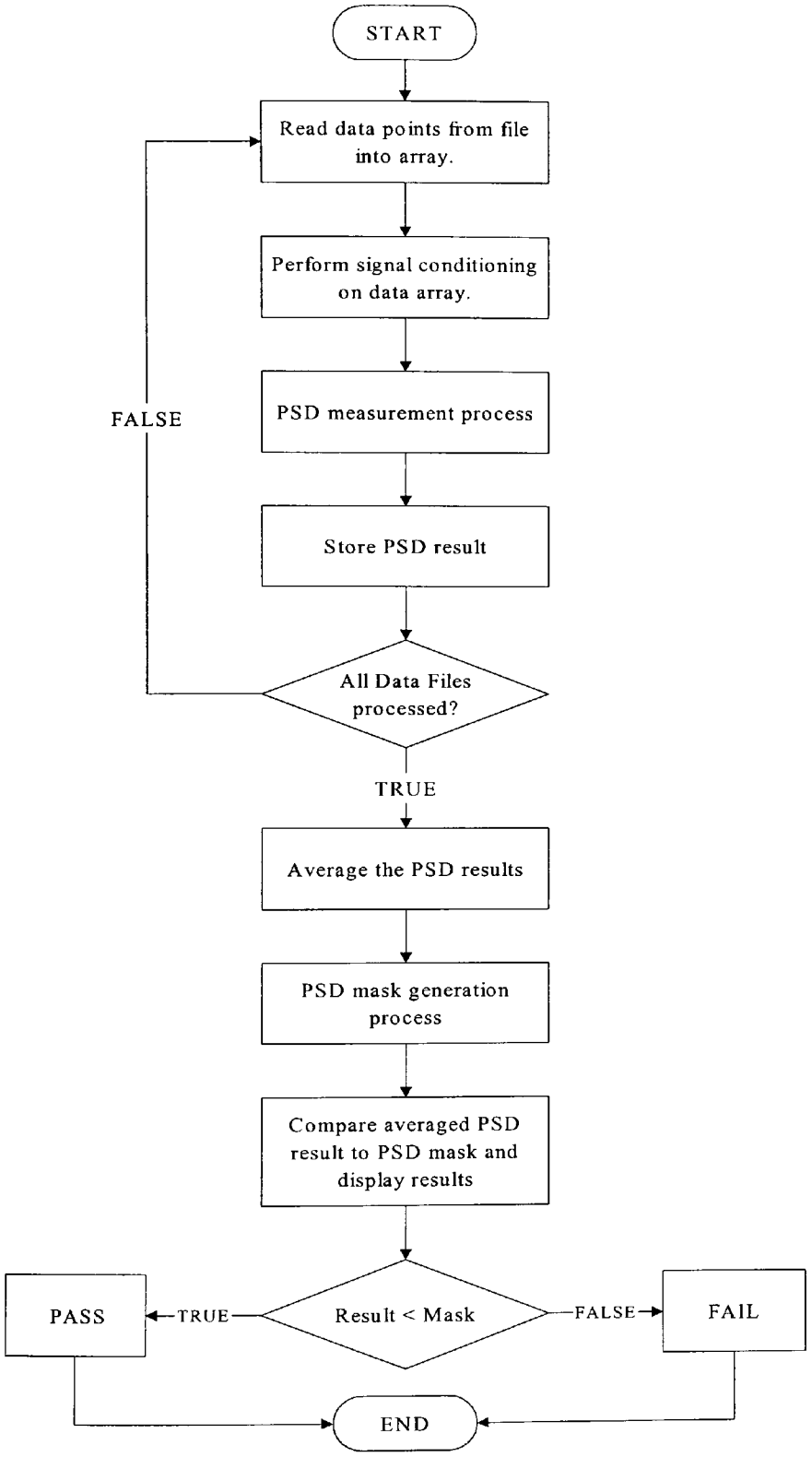


Figure 55 - Flowchart for the Periodogram V.I.

The read data from file into array process.

The sampled data from the unit under test is stored in comma-separated files on disk. The process loops through the file read process a number of user specified times. A control box labeled "Number of results to average" on the front panel specifies the loop number. As each successive file read process is completed the data is read from file, converted from text to integer and stored in an integer array.

The signal conditioning process.

The signal conditioning process converts from counts to peak voltage. The input voltage range of the ADC used in the hardware measurement system is 2V. As the ADC resolves to 12 bits this provides a maximum count value of 4096 where each count represents 488×10^{-6} volts. The data from the file is first referenced to the mid-count level by the subtraction of 2048 from each data value in the file. The data in counts is now multiplied by 1.6 to account for the transformer attenuation and by 3 to account for the attenuator. The result is a voltage representation referenced to zero volts.

The PSD measurement process.

The Calculate Spectrum VI performs the PSD measurement process. A software flowchart for the process is shown in figure 56. The process is performed in two stages. The first stage involves windowing the voltage data from the signal conditioning process by a window selected via a menu on the front panel. The system uses a Triangle, Hanning, Hamming, Blackman, Blackman-Harris or Kaiser-Bessel window in the data windowing process. The coherent gain value of the associated window is used to compensate for the processing loss induced by the windows. The

resulting windowed array of values is divided by the coherent gain value before the PSD is calculated. The process employs the National Instruments Auto Power Spectrum V.I. to calculate the single sided power spectrum. The PSD is calculated using the following equation and the result is in V^2_{RMS}.

$$\frac{FFT(signal).FFT*(signal)}{N^2}$$

(*signal*) in this case is the data in volts returned by the signal conditioning process. *FFT*(*signal*) is the FFT if the signal. *(*signal*) is the complex conjugate of (*signal*) . *FFT**(*signal*) is the FFT of *(*signal*). N is the number of data points and is 512 in this case. The voltage waveform was measured across a 135-ohm resistor and so the resulting power spectrum is divided by 135-ohms. This produces a result with units of Watts and is divided by 0.001 to reference the measurement to mili-Watts. The result in Watts is divided by the product of the FFT frequency resolution and the selected window ENBW value to produce a measurement of power over frequency. The final result is produced in units of decibels referenced to one mili-watt per Hertz or dBm/Hz.

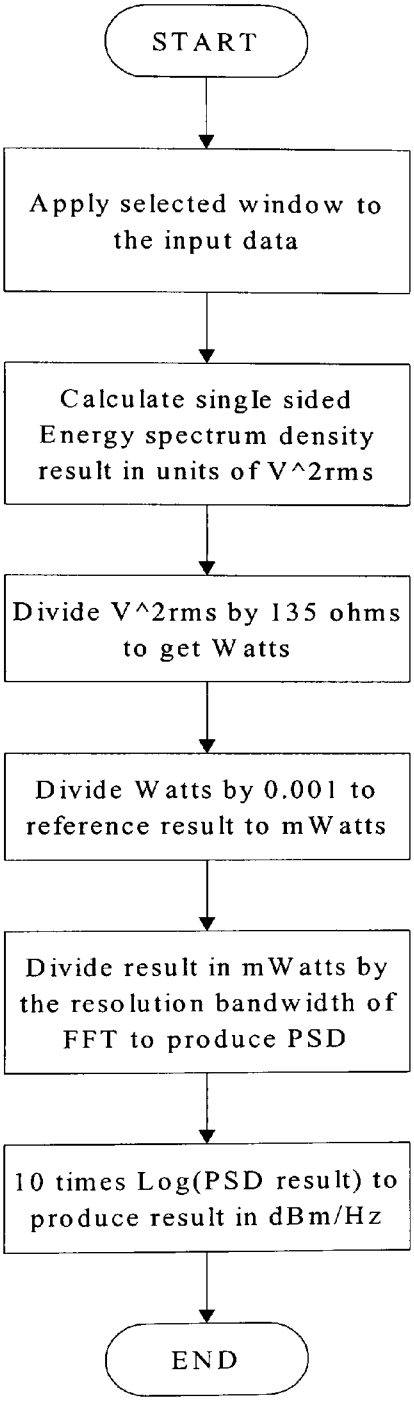


Figure 56 - Flowchart for the Calculate Spectrum V.I.

Average all PSD results process.

Once all the result files have been processed a two-dimensional array of PSD results is available. These are averaged to produce the final result. The final result is stored in an array containing 256 values. This is written to a comma separated text file.

PSD mask generation process.

The PSD mask is generated based on the mask details in the T1.601 standard. Facilities to generate the PSD mask as defined by the T1E1.4 draft spectrum management standard have are not provided by the software in this case. The PSD mask defined by T1.601 is still widely used in power limitation of ISDN equipment and provision for new PSD masks can be easily incorporated as software upgrades. The PSD mask as defined in T1.601 is presented in figure 57. The frequency range of interest is 0Hz to 256kHz. An array of 256 mask values is required.

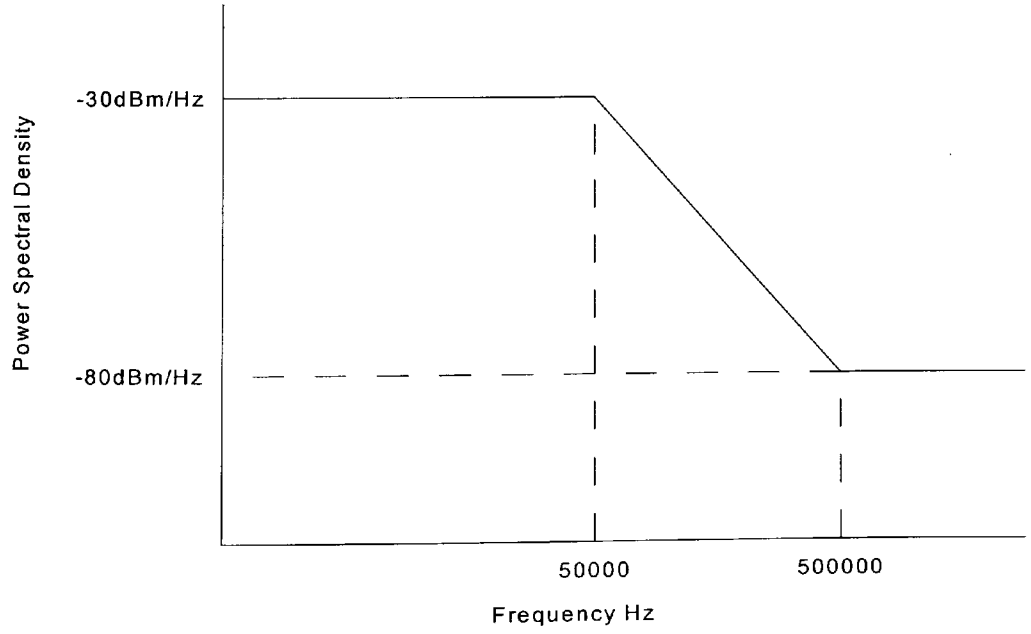


Figure 57 - The PSD mask for Basic Rate ISDN.

From the graph in figure 57 it is clear that the PSD is –30dBm/Hz over the frequency range 0 to 50kHz. From 50kHz to 500kHz the equation of a straight line can be employed to generate the PSD for the corresponding frequency. The equation of a straight line is $y = mx + c$ where y in this case is the PSD, x is the frequency, m is the slope of the line and c is a constant representing the intercept point of the line with the y-axis. The slope is found by the following.

$$slope = \frac{y_2 - y_1}{x_2 - x_1} = \frac{80 - 30}{500000 - 50000} = -111 \times 10^{-6}$$

It can be seen that the slope of the line is negative. The constant c is found by substitution as follows.

$$30 = -111 \times 10^{-6} \times 50000 + c$$

The constant c is found by manipulation of the above equation and is 24.5. The equation for the PSD mask values from 50kHz to 500kHz is thus,

$$PSD_{mask} = 111 \times 10^{-6} \times f + 24.5$$

The term PSD_{mask} is the PSD mask value and f is its corresponding frequency. The mask values are generated for frequency steps of 1kHz. The mask values are also written to a comma separated text file.

The Comparison of the averaged PSD result with the PSD mask.

The array containing the averaged PSD result is compared to the array containing the PSD mask. The comparison is performed on an element-by-element basis. The PSD results should be less than the mask values. If a PSD result element is less than the corresponding PSD mask element a count sequence is advanced. At the end of the comparison the count sequence should equal the number of elements in the array. If this is the case the pass/fail color indicator is set to green indicating a pass condition. Otherwise the pass/fail color indicator is set to red indicating a fail condition.

Chapter 5

5 The results and discussion.

The objective of the test is to determine if the transmitted power from the DUT is less than the PSD mask. The accuracy of the amplitude measurement is important over the frequency range from 0Hz to 80kHz as most of the signal power will be concentrated here. Signal power is expressed over a resolution bandwidth of 1kHz and has the units of dBm/Hz. The maximum transmitted signal power is defined by the PSD mask and is –30dBm/Hz over the frequency range from 0Hz to 50kHz. A value of –29dBm/Hz over this frequency range would be too high, and would be regarded as a fail. The 2B1Q reference signal transmitter filters its transmitted signal sharply above 240kHz and so it is not expected to observe significant power here. With reference to figure 6 in the portfolio overview, the signal power above 80kHz is not expected to be greater than –70dBm. The calculated PSD results from the software measurement programme are manually inspected to determine the spectrum characteristics. The pass or fail indication is produced by the software and is validated by the inspection process.

A reasonable level of confidence in the stability of the measured PSD estimate is required before scrutiny of the result can commence. Averaging of the measured spectra will reduce the amplitude variance of the result. As the number of averages taken is increased the amplitude variance should decrease to the point where further averaging does not provide a noticeable increase in spectral fidelity. Figure 58 shows the PSD result calculated from just one 512 point sample of the reference signal. As expected, the result shows poor spectral fidelity and further averaging is required.

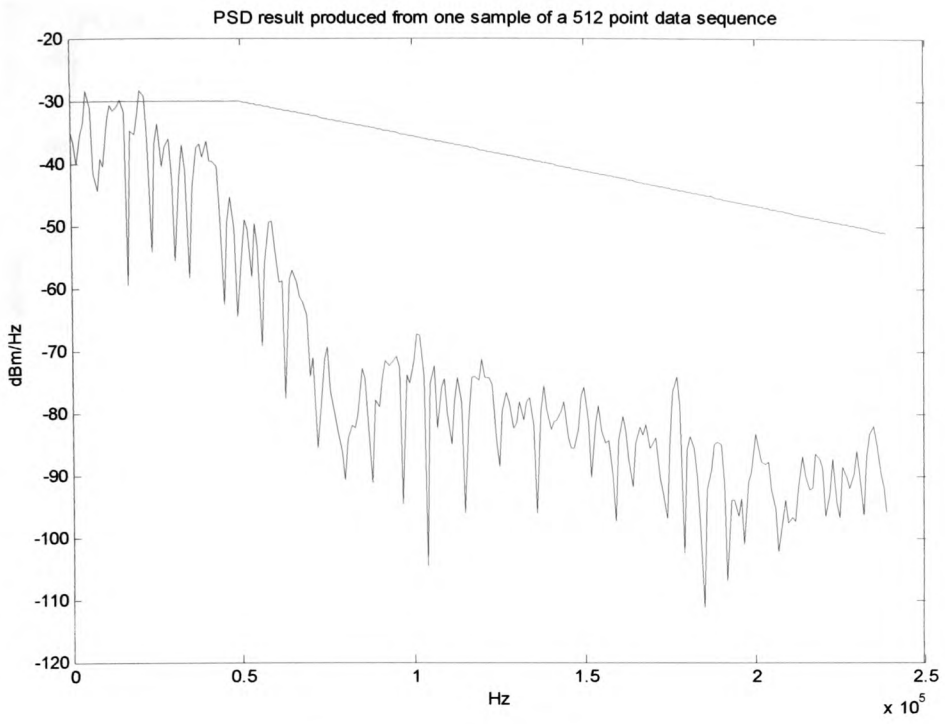


Figure 58 – PSD result calculated from one 512 point sequence.

Figure 59 shows the results of averaging successive PSD computations numbering from 50 to 100. It was found that the variance between successive PSD computations did not decrease dramatically beyond 70 averages. The final result was computed from the averaging of 100 successive PSD measurements containing 512 points of data. The results presented in the following section are all derived from an average of 100 PSD computations.

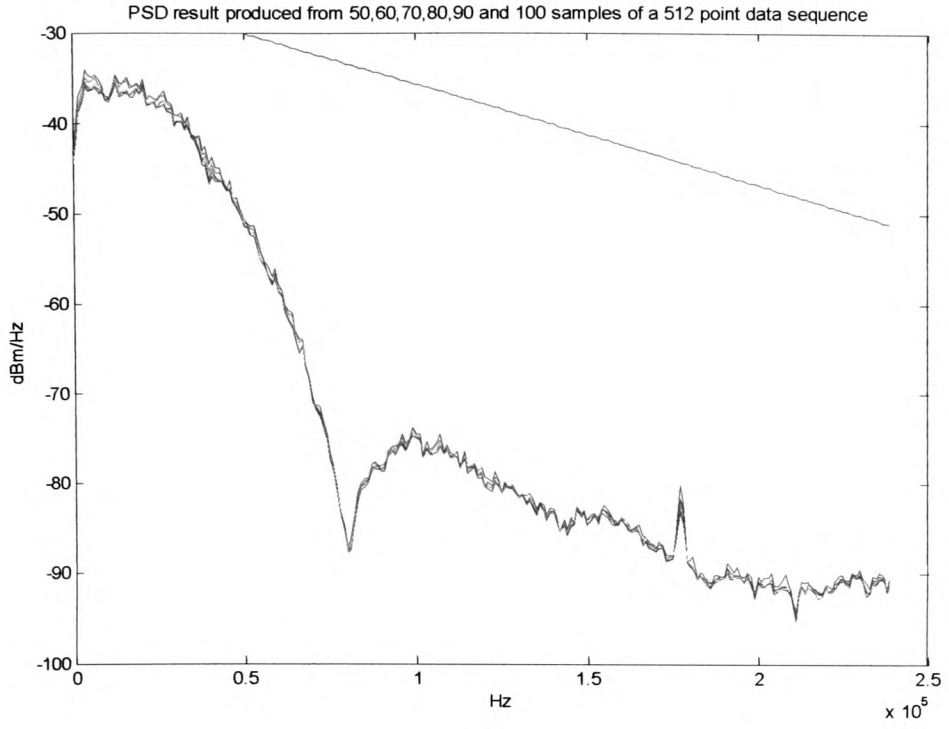


Figure 59 – The PSD results over 100 averages.

The effect of windowing on the PSD result.

Different window functions were applied to the sampled data sequence prior to PSD calculation and the following is a comparison of the results. The results in figures 60 through 62 show the PSD spectra derived from Hanning, Blackman, and Triangle windows. The Blackman-Harris window is one of the best window choices according to Harris [10]. Figure 63 compares the Blackman-Harris window to those already presented. The different results have been overlaid here and it can be seen that all the windows produce similar results.

The result using a Hamming window, shown in figure 64 does not mark the frequency null at 80kHz in the manner pronounced by the windows already mentioned. However the amplitude of the first spectral lobe from 0Hz to 80kHz and the amplitude of the second frequency lobe from 80kHz to 160kHz report similar

amplitude levels as the other windows. The frequency null at 80kHz is a characteristic that makes visual identification of the 80kbaud 2B1Q PSD easier. A test engineer would immediately look for this characteristic when analysing Basic-Rate ISDN spectra and a window that detracts from this would not be well employed in this application. Figure 65 demonstrates the visibly marked frequency null at 80kHz in comparison to that of the Hamming plot shown in red. The effect of Kaiser-Bessel windows with beta values of 2 and 3.5 has similar effects on the measured spectra and this is shown in figure 66. The peak amplitude of the PSD frequency lobes is consistent with the PSD amplitudes produced with the other windows. However, the frequency null at 80kHz is not as pronounced.

Figure 67 compares the Kaiser-Bessel windows (plotted in green) to the results attained with the other windows. Windows that enhance the characteristics of the spectrum are the best choice in this case. It is important that the reported amplitude is correctly presented, as this is the main criterion used by the software in determining a pass or fail. It can be seen that all the windows report the same amplitude level and are thus validated for use. The Hanning, Blackman, Triangle and Blackman-Harris windows gives the best enhancement to the spectral characteristics while reporting the correct amplitude levels. Any one of these windows are valid for use in this application however, the Triangle window is considered the best choice due to its simplicity.

All the results presented in this chapter are consistent in reporting amplitude and match that measured with a spectrum analyzer during validation exercises. The PSD measurement process was replicated using a programme written in the Mathlab simulation package. The PSD result produced with the Labview measurement programme was compared to the PSD result produced by the Mathlab. This was

performed to validate the integrity of the Labview PSD measurement programme. A comparison between the two results was made. Figure 68 shows the result. The measurement process employs similar windows in both cases. There is a difference of 0.5dB approximately across the frequency range in question. The difference between the two may be attributed to the difference in window coefficients between those generated by the Mathlab implementation and that of Labview. The window coefficients generated by the labview window functions cannot be examined, as these are primitive blocks. However, the comparison between the spectral characteristics of both implementations is very similar and this is true for all cases.

It has been concluded that the integrity of the results are good. The spectral characteristics of the measurement in figure 68 show a frequency null at 80kHz as expected. The reference signal transmitter filters the transmitted signal sharply above 160kHz and it is not expected to observe much signal power at frequencies above this, which was observed to be the case. The amplitude of the PSD above 50kHz shows the amplitude to be less than –70dBm as expected. As frequency increases, the amplitude profile suggests than a frequency null at 160kHz exists. This is not seen in the calculated spectra, as noise is present. The peak at 175kHz is not a component of the signal generated by the reference signal transmitter and is recognised as noise. The PSD result in this case is compliant with the PSD mask as defined in the T1.601 standard. The Labview software successfully produced the PSD graphical result along with a pass indication.

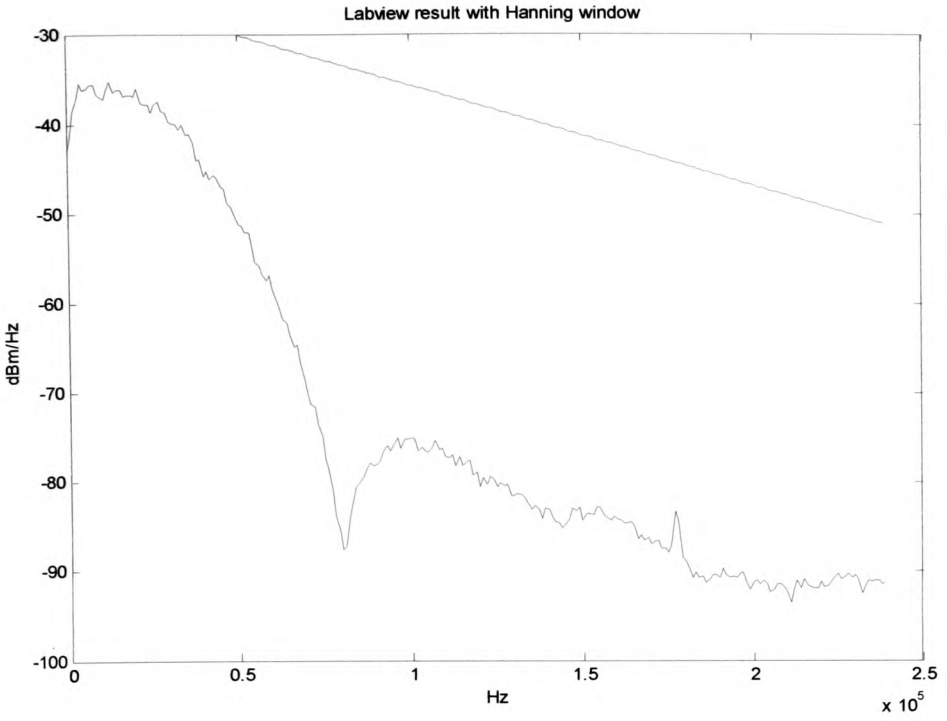


Figure 60 - The PSD spectrum with a Hanning window.

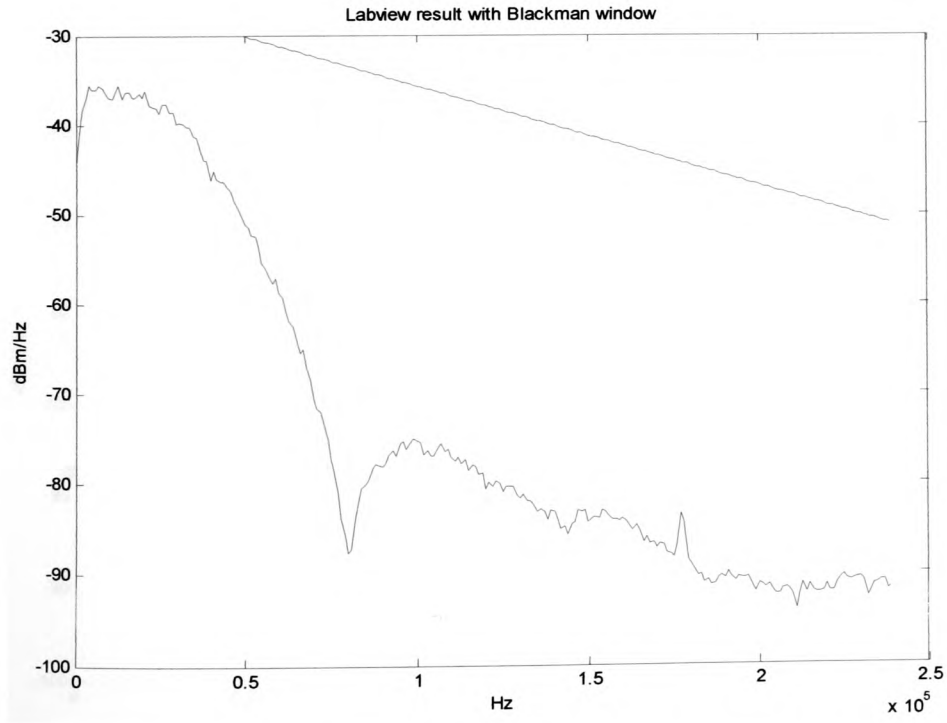


Figure 61 - The PSD spectrum with a Blackman window.

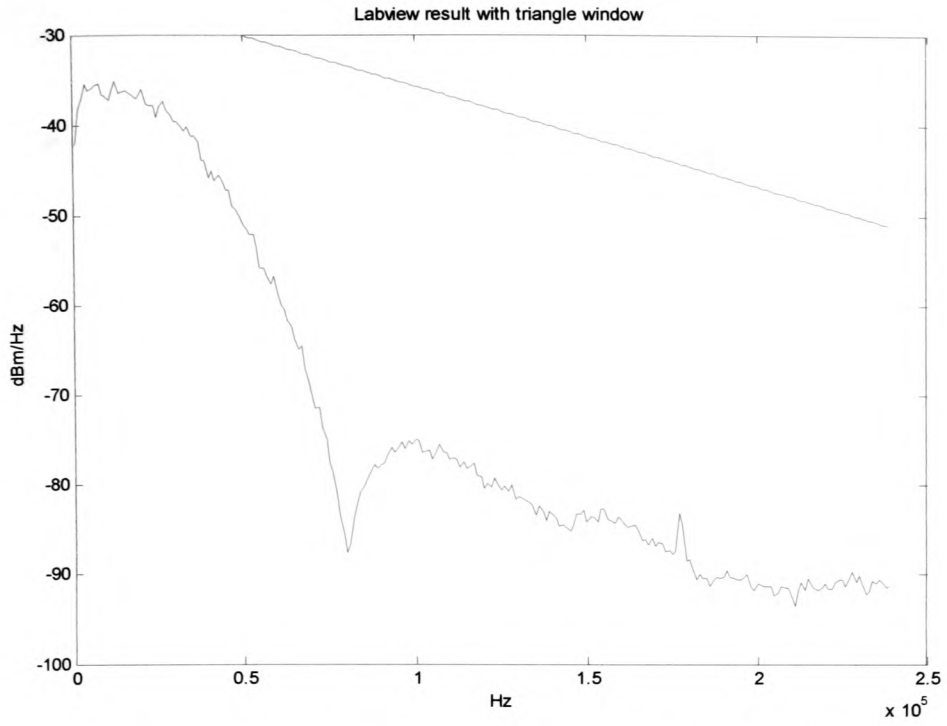


Figure 62 - The PSD spectrum with a Triangle window.

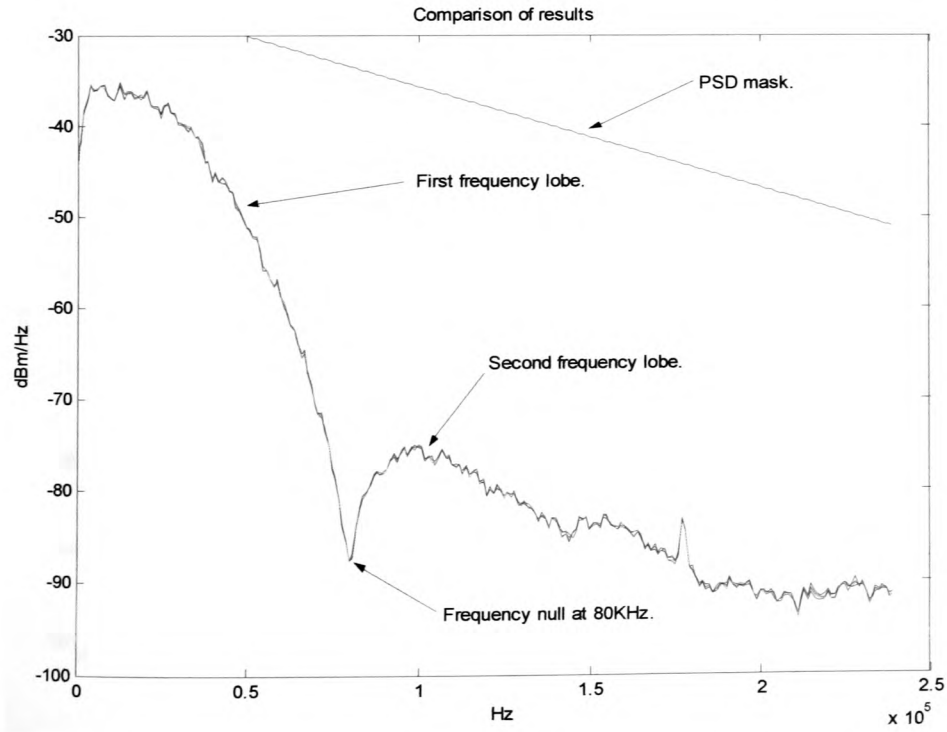


Figure 63 - A comparison with the Blackman-Harris window.

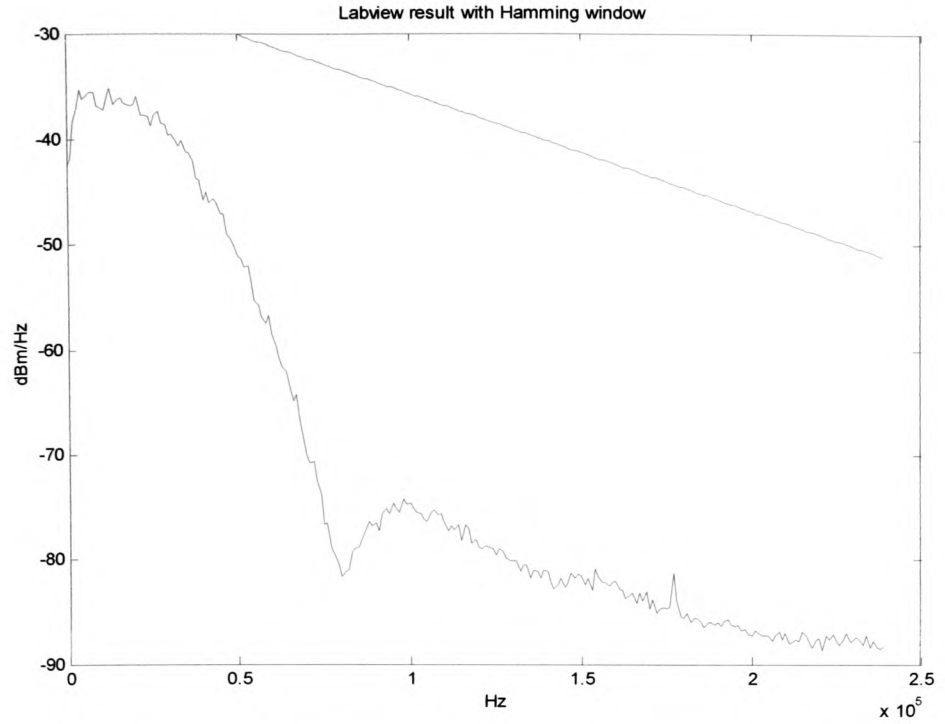


Figure 64 - The PSD spectrum with a Hamming window.

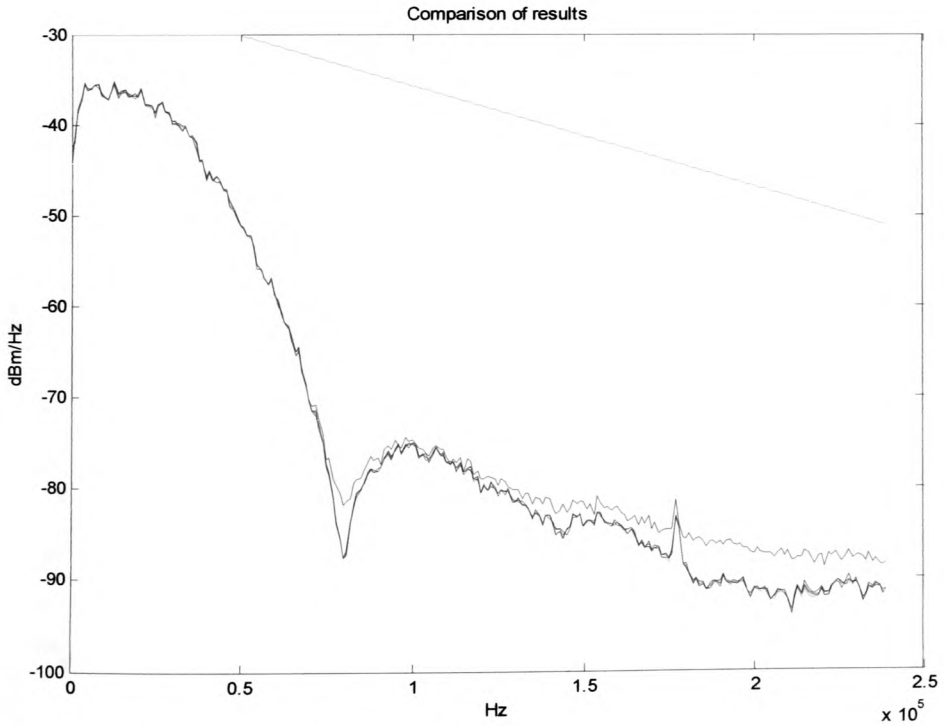


Figure 65 - The red plot shows the PSD computed with a Hamming window.

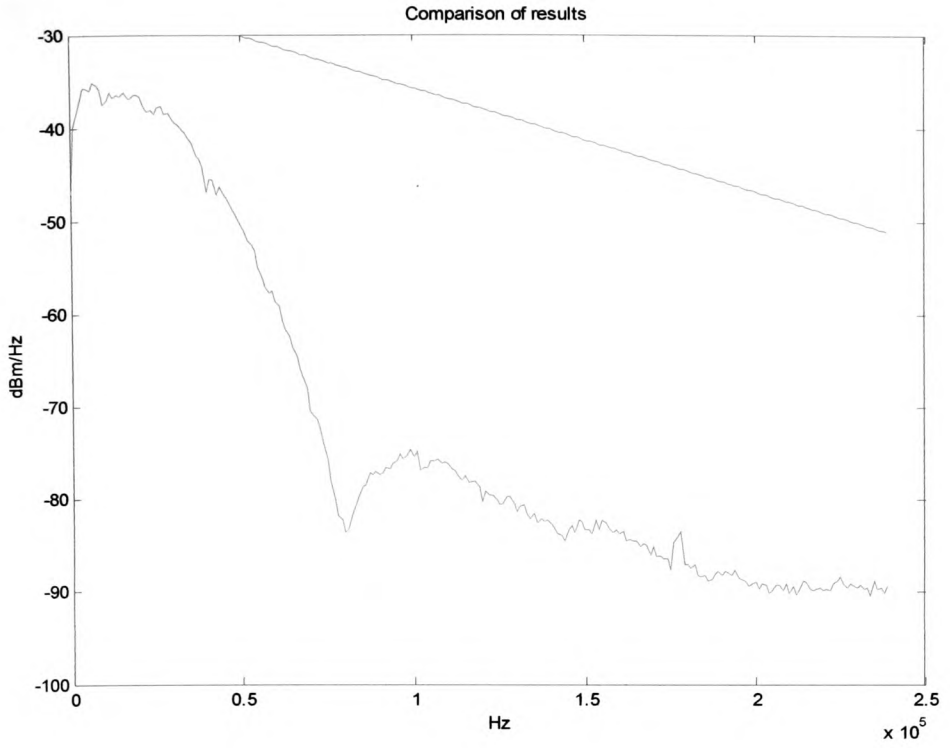


Figure 66 – PSD computed with Kaiser-Bessel windows with $\beta = 2$ and $\beta = 3.5$.

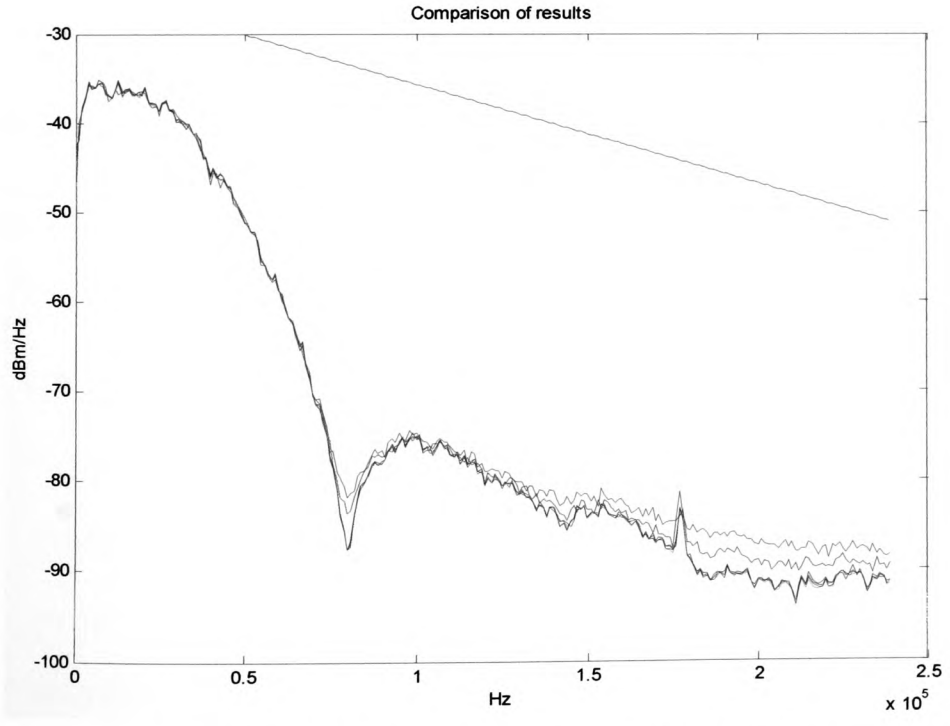


Figure 67 - Comparison of the PSD computed with the different windows.

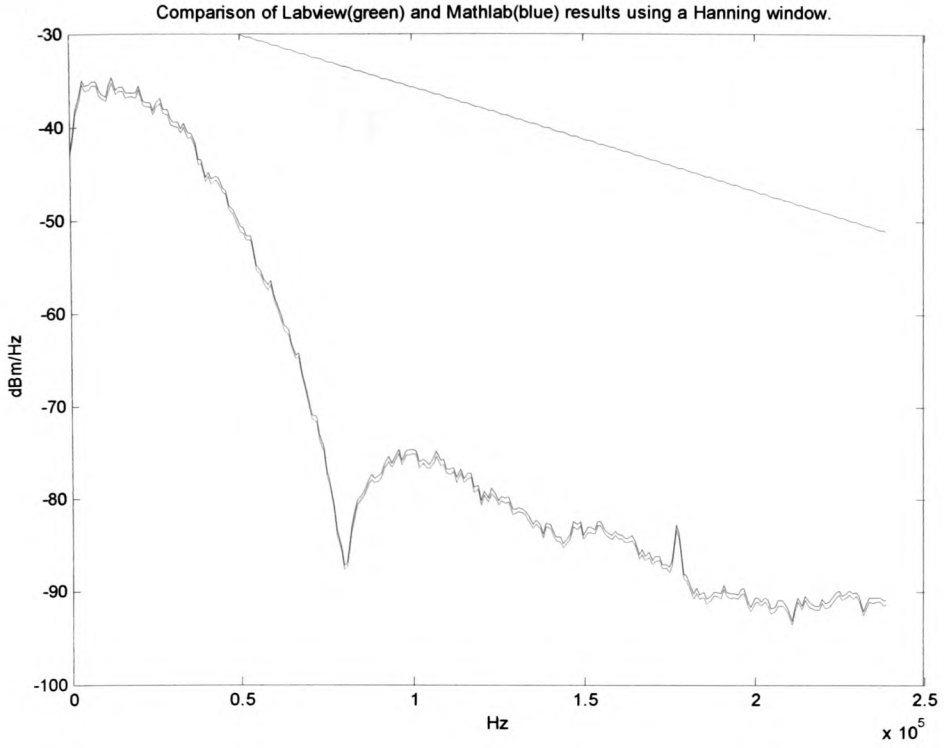


Figure 68 - Validation of Labview PSD result with Mathlab.

Chapter 6

6 Conclusion and Further work.

The aim of the project was to design a system to measure PSD for Basic-Rate ISDN applications and facilitate PSD compliance testing of ISDN equipment. The objective was to produce a flexible design that could be easily modified in the future to measure the PSD of data rates above the Basic-Rate. The following conclusions can be drawn from the work.

6.1 Conclusions.

1. The results presented in chapter 5 show that this objective has been achieved. The characteristics of the spectrum are clearly identified as those of a 2B1Q signal operating at 80kbaud. The Measurement software provides a clear indication of a pass or fail condition when a simple test indication is required and offers a user-friendly intuitive environment running on Microsoft Windows. The graphical display of the PSD result compared with the PSD mask allows an Engineer to observe the test result in more detail. A PSD mask can be easily generated for other test classes and incorporated within the test system as a software upgrade. The test results can be written to file and stored on the hard disk for analysis thus meeting all the requirements identified for the measurement system.
2. The FPGA interfaces synchronously with the hardware data-acquisition system. In the current implementation the sampling clock is generated by the FPGA and is defined at the design stage. Redesigning the FPGA to produce a

different sampling clock allows flexibility over the frequency range displayed by the PSD measurement software. The possibility also exists to design an FPGA firmware solution that allows the sampling rate for data-acquisition to be selected via a register value in the FPGA. A variable clock divider network could be used to produce a sampling clock derived from the 15.36MHz master clock. The register value could determine a clock divider ratio and thus allows a flexible sampling clock to be produced. The wide-ranging frequency spectrums required for PSD measurement of other test classes could be facilitated this way. The ADC chosen for this application supports the necessary sampling rate required for all the spectrum classes defined in the T1E1.4 standard. The FPGA implements an interface to the ADC and this interface can be easily redesigned should the need to employ a different ADC become necessary.

3. The FPGA currently employed (Xilinx 4010XL/Sparten) has the data storage facilities for 512 points of data. The FPGA can be upgraded to the next larger capacity member in the Xilinx 4K/Spartan family range. This could allow for the storage of more data points thus providing an increase in measurement frequency resolution. The Xilinx 4K/Spartan family of FPGAs has the same physical footprint as the Atmel family of 4K FPGAs. This provides a second source of components to ensure that component shortages or pricing does not threaten production schedules. The flexibility offered by FPGA technology in this application provides the ability to react to market demands rapidly thus satisfying the objective to produce a future proof solution.

4. The concept of future proof design using reconfigurable logic has not been well embraced by industry and the technology to realise this has only been

regarded as finically viable in recent years. The FPGA has been instrumental in providing increased circuit integration to reduce PCB space and competes well with an ASSP in small volumes. However, the application of FPGA technology to product designs where different FPGA configurations give a single printed circuit board totally different functionality is new. This potentially allows a manufacturer to incorporate the functionality of a number of products using one PCB. The ability to change the product functionality as the market demands progress provides distinct advantages over competitors.

5. There is no current solution on the market that provides the functionality demonstrated by this project. Products do exist that provide cable evaluation and PSD measurement facilities. However, these solutions are expensive and do not provide all test facilities required for DSL commissioning. Laptop computers and Palm computers have seen wide deployment in the telecommunications industry. Modern computers can easily provide the performance to facilitate the software analysis and DSP required to test and commission digital telephone systems. The development of a hardware measurement tool that can be reconfigured to provide different test facilities is a considerable contribution to Industry.

6.2 Further work.

The project work presented here demonstrates how to measure the PSD of a Digital Subscriber Line signal. The hardware and software was successfully designed to implement the PSD measurement feature and the flexibility of the design was demonstrated. The following areas were identified where further work would provide an improvement over that which has already been presented.

1. The PSD was calculated by averaging 100 consecutive spectra derived from 512 points of data. The reduction in variance of the spectra using this technique was concluded to be adequate for this application. However, if it is required to plot the PSD over a frequency range beyond 256kHz then 512 points of data will not be adequate to maintain a resolution of 1kHz. In this case it would be necessary to upgrade to a larger capacity FPGA and employ more RAM.

2. According to the literature review presented in the Portfolio overview, the variance in the calculated spectra can be further reduced using the Welch method of spectrum estimation. In this case a sampled sequence is broken down into several overlapping data sequences, the PSD of each sequence is calculated and then averaged to yield the final result. The lack of RAM facilities offered by the Xilinx 4010 FPGA prevented further investigation along this line. The current PCB will allow the population of a larger capacity FPGA and the modular design of the FPGA firmware supports RAM expansion.

3. The DSP functions required to perform the PSD calculation were implemented in software. The current design requires a computer to retrieve the measurement samples from the FPGA in order to calculate the PSD. This implementation may not be suitable for portable equipment. The software may be implemented to run on a Microcontroller. However, a high performance Microcontroller with DSP features such as integrated FFT hardware would be required. The integration capabilities and high performance of modern FPGAs support the development of complex DSP functions. The FPGA design could therefore be expanded to include the PSD measurement function.

References.

[1] Draft American Standard under review by the American National Standards Institute, Spectrum Management For Loop transmission Systems, T1E1.4/2000-002R2

[2] Texas Instruments, AFE1224 2Mbps Analogue Front End, Data sheet number PDS-1548A.

[3] International Telecommunications Union (ITU-T), Recommendation O.150, Specifications of Measuring Equipment, 1996.

[4] American National Standards Institute, Integrated Services Digital Network (ISDN) – Basic Access Intreface for use on Metallic Loops for application on the Nework Side of the NT (Layer 1 specification), T1.601-1992.

[5] Texas Instruments, INA121 Differential Amplifier, Data sheet.

[6] National Instruments, DAQ CARD-1200 manual.

[7] Texas Instruments, ADS807 12 bit, 53MHz Sampling Analog to Digital Converter. Data sheet number PDS-1396C.

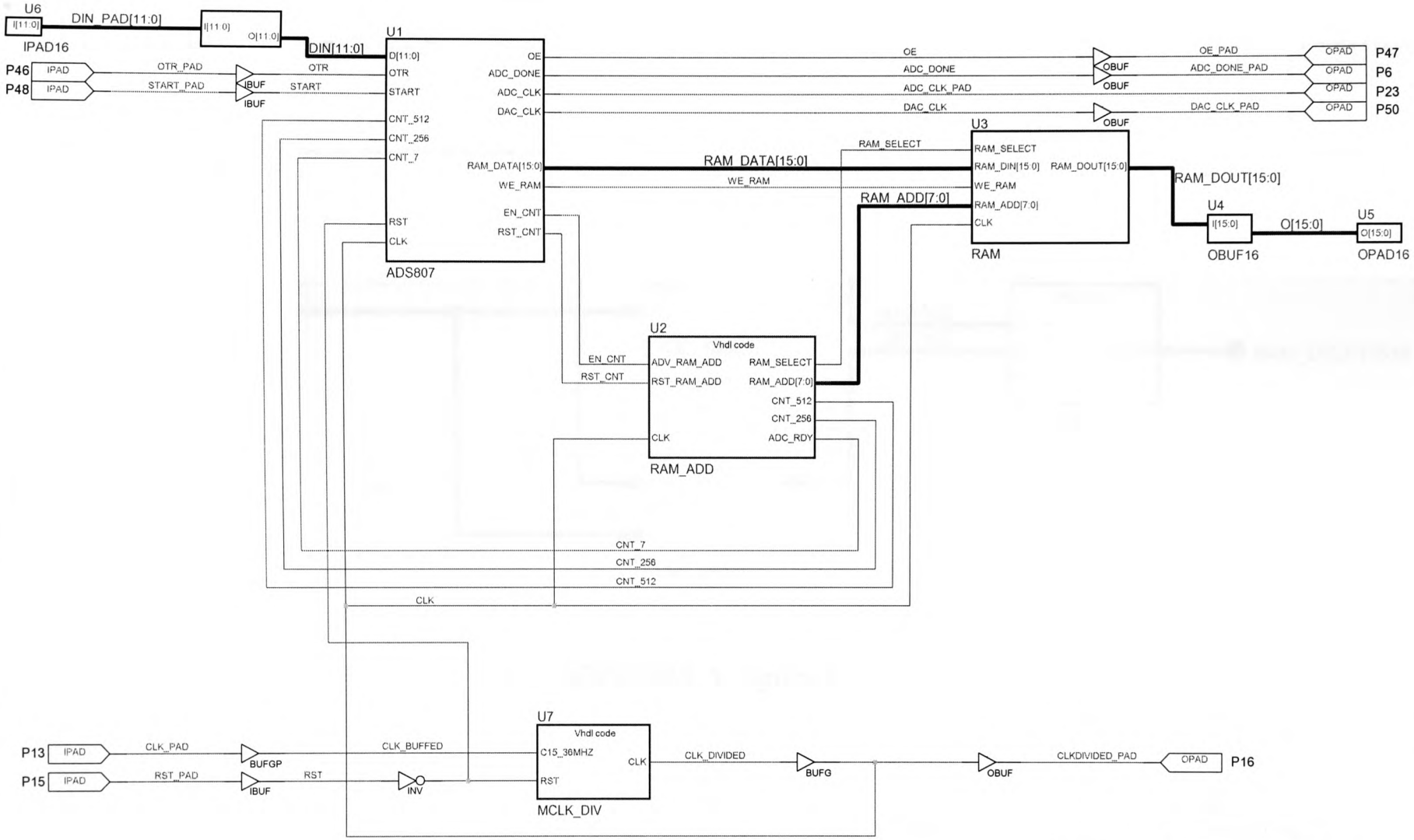
[8] Texas Instruments, OPA2682 Dual, Wideband, Fixed Gain Buffer Amplifier with disable. Data sheet number PDS-1477B.

[9] Michael Jacob, Applications and Design with Analogue Integrated Circuits, Regents/Prentice Hall, ISDN 0-13-032145-1.

[10] Harris F.J. On the use of Windows for Harmonic Analysis with the Discrete Fourier Transform, Proceedings of the IEEE, Vol.66, No. 1. Jan. 1978

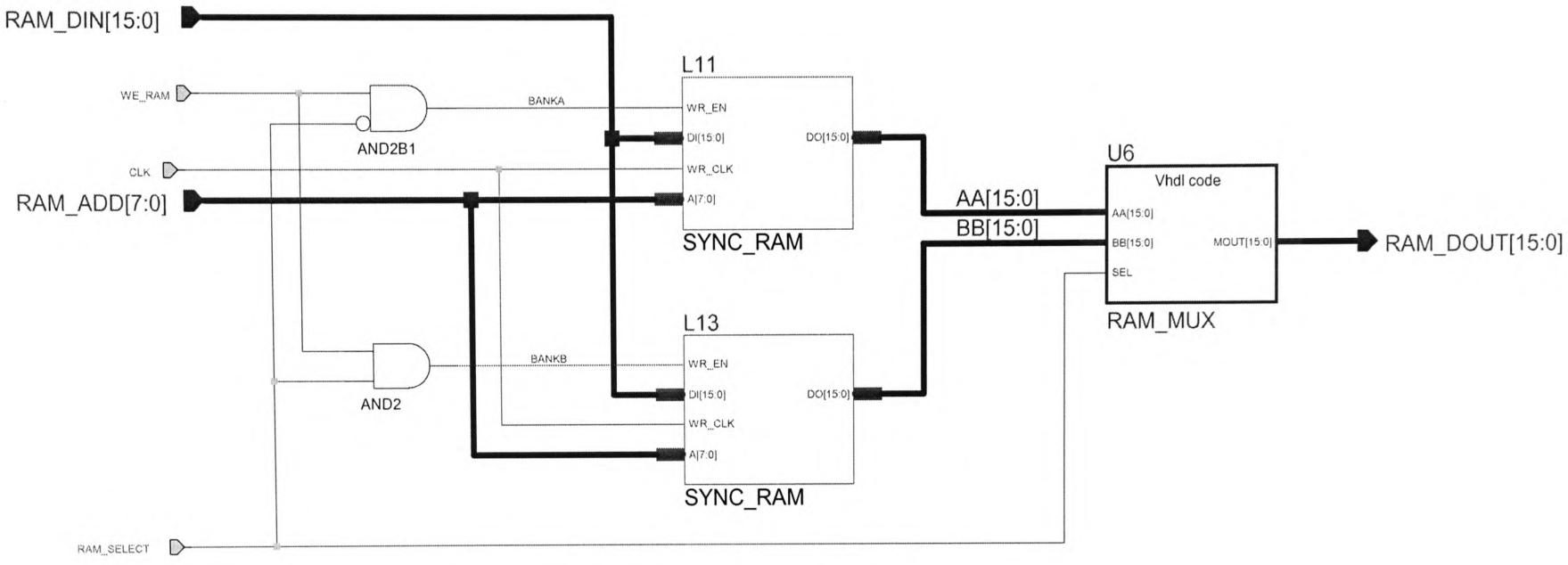
Appendix A

The FPGA firmware block diagrams



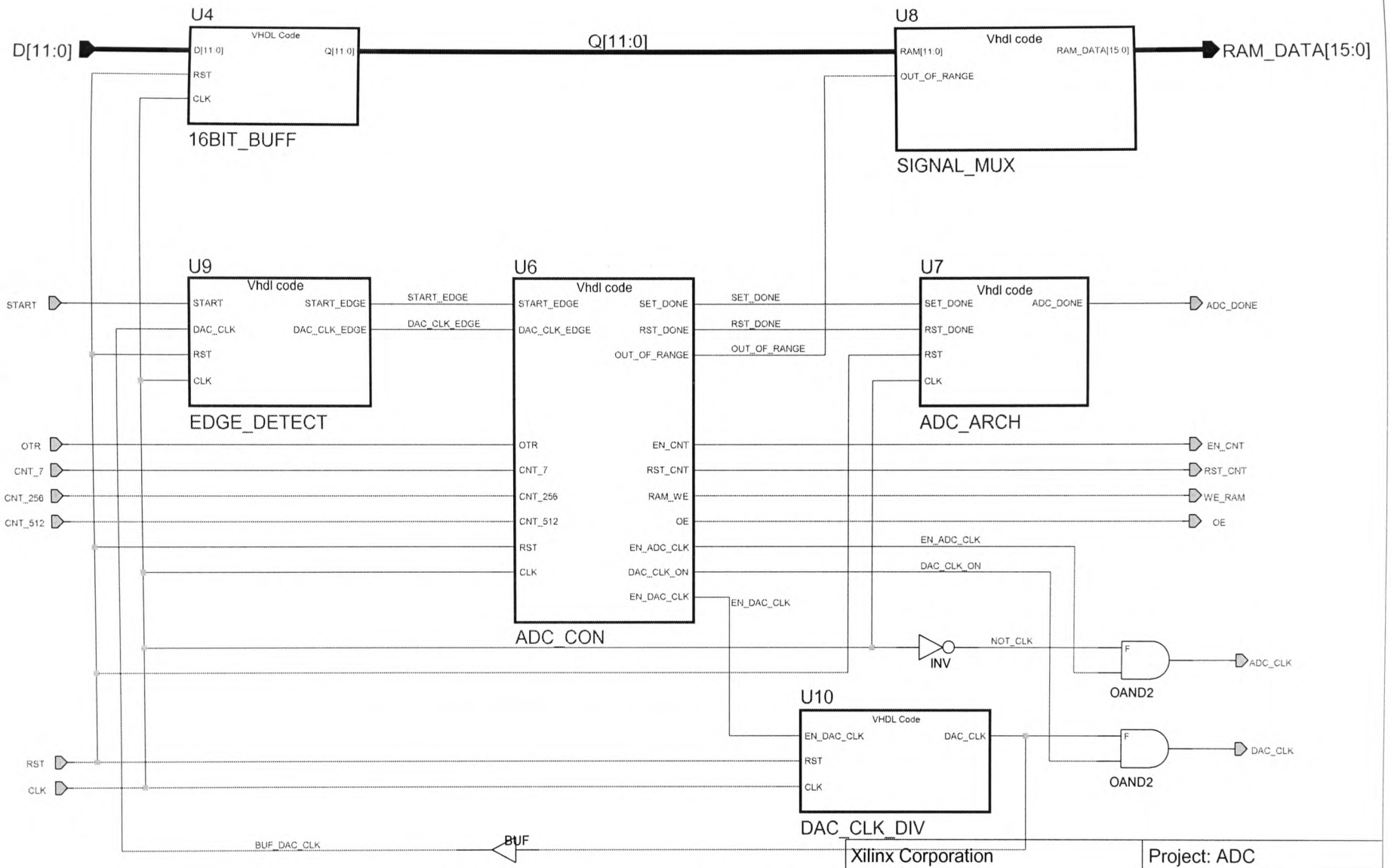
APPENDIX A Figure 1

Xilinx Corporation	Project: ADC
2100 Logic Drive	Macro
San Jose, CA 95124	Date: 5/23/02
Date Last Modified:	



APPENDIX A Figure 2

Xilinx Corporation	Project: [None]
2100 Logic Drive	Macro
San Jose, CA 95124	Date: 5/23/02
Date Last Modified:	



APPENDIX A Figure 3

Xilinx Corporation	Project: ADC
2100 Logic Drive	Macro: ADS807
San Jose, CA 95124	Date: 5/23/02
Date Last Modified: 1/9/4	

Report Number 2.

A Bit Error Rate Test System for Basic Rate ISDN.

Patrick Sugrue

Report number 2 is part of a portfolio of projects and is submitted in partial fulfilment of the requirements of the University of Glamorgan for the degree of Master of Philosophy.

15-December-2003.

Table of contents.

Appendix

List of figures.

List of tables.

Abbreviations.

ADPLL	All digital phase locked loop
AMI	Alternate mark inversion.
ASM	Algorithmic state machine.
ASSP	Application specific standard products.

BERT	Bit Error Rate Testing.
CLB	Configurable logic blocks.
FPGA	Field programmable gate array.
IOM-2	ISDN orientated modular inteface – revision 2.
ISDN	Integrated systems digital network.
ITU-T	Telecommunications sector of the International telecommunications union.
NT	Network Terminator.
ppm	Parts per million.
PCB	Printed circuit board.
PLL	Phase locked loop.
PRBS	Pseudo-random binary sequence.
TE	Terminal equipment.
VHDL	Very high speed integrated circuit hardware description language.

1. Introduction.

This report focuses on the design and implementation of hardware components to facilitate Bit Error Rate Testing (BERT) on the basic-rate integrated systems digital network (ISDN). The main design effort concentrates on the circuitry required by the ISDN Terminal equipment (TE) to establish a loopback at the Network Terminator (NT), monitor a pseudo-random sequence (PRBS) sent to the NT and count bit errors in the looped back PRBS over the test duration. The analogue components to allow interfacing to the ISDN S bus are also designed and an NT design is realised with the functionality to facilitate test and validation. The design work employs a top down synchronous design technique using the Very high speed integrated circuit hardware description language (VHDL). Xilinx Field programmable gate arrays (FPGA) are targeted at the implementation stage. Figure 1 presents a typical BERT test setup. The S bus is a four wire transmission system connecting the NT to each TE in the network. Two wires are used for each direction of data transmission as shown. Differential signaling is employed and the signal components in each differential signal will be referred to as signal A and signal B throughout this report.

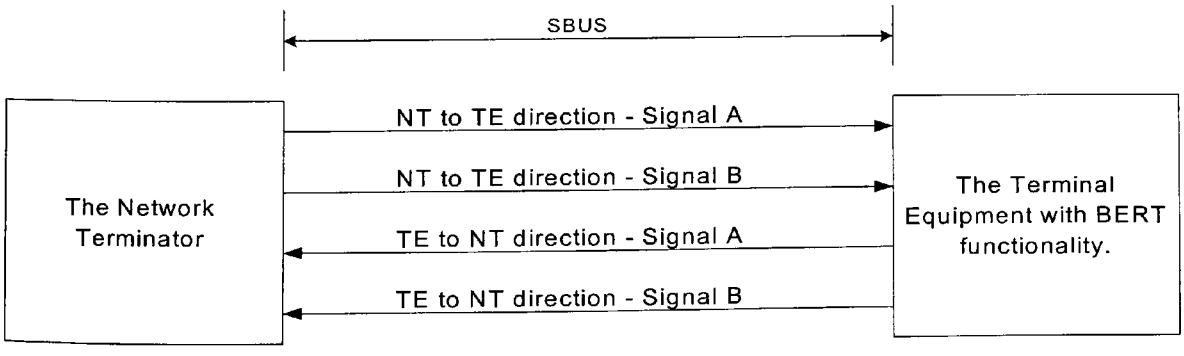


Figure 1 - The test and verification set-up.

The BERT system is used to test the transmission line between the NT and a TE. The BERT system can be segregated into three distinct functions. These functions are synchronisation with the ISDN S bus, synchronisation with the maintenance channel to request a loopback and Bit error rate testing to check the transmission capability of the system. The system overall objective is to provide a solution that achieves galvanic isolation with the S bus and realisation of the electronics to achieve the above functions. The design of an NT will also be presented to provide a test environment to enable validation of the design.

1.1 The problem with current BERT designs.

BERT systems require an ISDN S bus transceiver chip to facilitate ISDN signal transmission. Currently available ISDN S bus transceiver chips provide integrated maintenance channel controllers. The analogue circuitry required to transmit and receive the ISDN signals is also included in these chips. Pseudo-random binary sequence (PRBS) transmitter and receiver circuits are essential to facilitate BERT testing. However, ISDN transceiver manufacturers do not provide integrated PRBS circuits with their products. This forces designers to realise PRBS functionality in one of three ways.

Firstly, application specific standard products (ASSP) are available with standard PRBS circuits required for telecommunications testing purposes. However, these chips are not specifically designed to interface with basic rate ISDN chip sets. This makes the use of ASSPs difficult to integrate within designs. Also, the adoption an extra chip is expensive and places an extra burden on designers challenged with fitting the electronics in the restricted space defined by telephone

handsets. It is desirable to achieve a high level of circuit integration and maintain the required components at a minimum. The volumes associated with the production of ISDN test equipment are low. Thus the incentive for integrated circuit manufacturers to integrate PRBS hardware within their designs is undermined. Secondly, it is possible to implement the PRBS functionality in software. However, this places extensive overhead on microprocessor resources and would be better suited to hardware implementation. Thirdly, and FPGA can be employed to implement to PRBS test functionality. The FPGA can be interfaced to the ISDN chip set via the industry standard ISDN orientated modular inteface – revision 2 (IOM-2) [1]. This is the most widely adopted method. However, this method is similar to the first, the FPGA being the extra chip required in this case. If the ISDN chip set does not support the IOM-2 bus then interfacing may be difficult.

Field programmable gate array (FGPA) technology offers a solution to this problem. The circuit integration capabilities of modern day FPGAs can easily support ISDN feature requirements. The ability to reconfigure the FPGA allows designers a design alternative to ASICs that is more flexible, cost effective for low volumes and more conducive to meeting the reduced time to market requirements demanded by industry. Figure 2 proposes how an FPGA could be integrated within a typical TE design. The design is composed of an analogue signal receiver, an analogue signal transmitter and the FPGA, which can be employed to integrate the digital electronics to perform ISDN operations. The integration capability of the FPGA would allow test equipment manufacturers to develop test solutions that do not rely on ASSPs for realisation of BERT features. The FPGA design approach also allows extended product life cycle as designs may be adapted as the market changes.

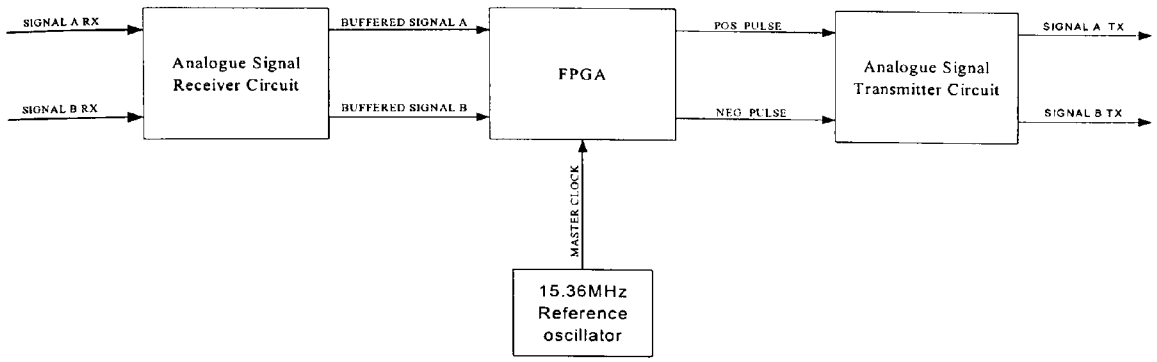


Figure 2 - The top-level system block diagram.

1.2 The project objectives.

The following list objectives were identified for project number 2.

1. To investigate the integration capabilities of FPGAs with application to basic rate ISDN BERT functionality
2. To challenge the design philosophy currently adopted by test equipment designers by presenting an alternative design approach to currently adopted methods, which rely heavily on commercially available ASSPs to realise test functions.
3. To develop both TE and NT solutions with the circuitry required to facilitate BERT testing in accordance with recommendation I.430 produced by the Telecommunications sector of the International telecommunications union (ITU-T) [2].

1.3 Summary of achievements.

The project work succeeded in producing a TE and NT design with the capabilities to perform a BERT function. The NT was successfully designed and implemented and the required ISDN signaling, maintenance channel and loopback facilities were produced to validate the TE design. The TE design was successfully designed and implemented and the objectives to integrate the PRBS test circuits with the ISDN layer one functions were achieved. The FPGA was shown to be instrumental in the implementation of the BERT system and demonstrated that greater integration of test features for ISDN testing applications can be realised.

1.4 Report structure.

Chapter two presents the design of the analogue signal receiver circuit. The receiver circuit operation is described and oscilloscope plots are presented showing the signals in the system.

Chapter three describes the analogue signal transmitter circuit. A technical overview of the circuit design is provided and an oscilloscope plot of the transmitted signal is shown to validate the circuit design.

Chapter four outlines the FPGA design for the TE. An overview of the design is presented with a top-level block diagram. The design is organised as a hierarchical system composed of a number of sub-blocks. Each sub-block is described in detail. The synchronisation circuit and phase locked loop sub-block is described. A design verification procedure for the sub-block is presented and the measured results are validated. The ISDN signal coding circuits, the maintenance

channel synchronisation circuitry and the PRBS transmitter and receiver circuitry is described. Each functional block is described in detail.

Chapter five presents an overview of the FPGA design for the NT. The NT design is broken down into its sub blocks and a functional description of each block is given.

Chapter six presents the results of the design verification exercise. Oscilloscope plots present the signals measured during the procedure and FPGA implementation details are produced. The results are analysed and conclusions are drawn that validate the design.

Chapter seven presents the conclusions drawn from the work and future work is identified.

2 The analogue signal receiver circuit.

The analogue receiver circuit extracts the analogue signals from the S bus and performs the analogue to digital conversion to interfacing with the FPGA. The analogue receiver circuit is identical for both TE and NT cases. The analogue circuit design is best described with reference to the voltage levels at different points in the circuit. Figure 3 shows the voltage waveforms on the telephone line, at the output of the transformer and at the output of the buffering circuit. The worst-case signal attenuation in the NT to the TE direction occurs when a point-to-point configuration is used. Point-to-point configuration involves an NT connected to one TE with a telephone line that can run for a maximum distance of up to 1km. The attenuation in this case is specified by recommendation I.430 as 6dB [2]. The nominal amplitude level of the transmitted signal from the NT, as defined by the I.430 pulse mask is 750mV. The worst-case amplitude level is 675mV from the NT, which when subjected to an attenuation of 6dB would reduce the amplitude level to 337mV at the TE receiver.

The transformer ratio is 1:2 and a signal level of 750mV on the telephone line will be amplified to 1.5V on the isolated side of the transformer. The minimum amplitude level of 337mV representing a valid signal on the telephone line will be increased to 674mV on the other side of the transformer. Signals that fall below amplitudes of 250mV in this case are regarded as noise and are rejected by the signal receiver. The Alternate mark inversion (AMI) signal on the telephone line is differential and thus not referenced to any voltage point. It is therefore necessary to pull the received AMI signal to the mid-rail of the supply voltage prior to buffering. The supply voltage for the analogue circuit is 5V and a supply splitter employed to

produce 2.5V. The 2.5V level is used as the common mode operating voltage point for the receiver. The receiver AMI signal is pulled to alternate about this point labeled as COMM_RX on the diagram.

The FPGA circuit is supplied by a 3.3V supply and thus some buffering is required to compensate for the different voltage levels. The voltage level labeled as RX_THRES in figure 3 represents the level detection value, which must be satisfied before a received signal is considered valid. Figure 3 shows how the buffered signals are derived from signal A and signal B. T1 through T6 represent time in periods of 5.2μS. The three sets of signals all occur over the same time period. Due to the differential transmission technique Signal B is the exact opposite of signal A. The buffering circuit converts the differential analogue signal to two digital signals labeled as FPGA_RXA and FPGA_RXB. Figure 4 presents the timing diagram for the analogue receiver.

The timing diagram presents one frame of data from the NT to the TE with just the synchronisation pulses shown. The diagram shows how the differential signal is converted to a form suitable for interfacing to the FPGA. The differential signal is separated into signal A and signal B for clarity. The signal has two coding violations set by the L, F symbol pair followed by the L, Fa symbol pair. The signal FPGA_RXA passes the positive part of the differential signal to the FPGA and the signal FPGA_RXB passes the negative part. Using this technique the FPGA should only see one violation on FPGA_RXA and one violation on FPGA_RXB. The polarity of transmission is undefined. The FPGA firmware has to synchronise to the ISDN signal based on the relationship between these violations. The only fixed event from which to derive timing is the relationship between the F bit and the next balance bit L. The balance bit L will always be the opposite polarity of the F bit.

This coding rule feature will be used in the digital design to facilitate synchronisation and phase locking of the clocks.

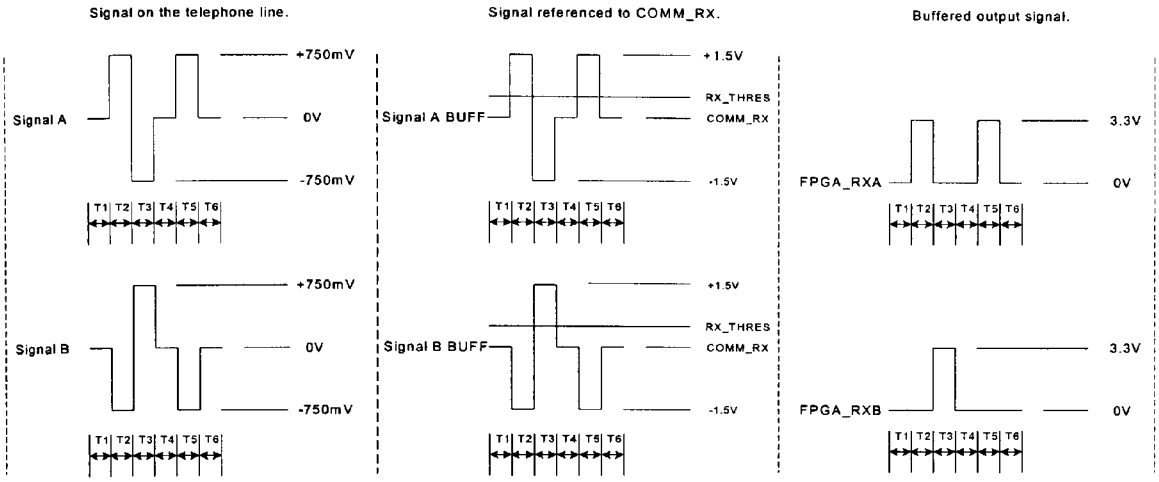


Figure 3 - Analogue signals levels.

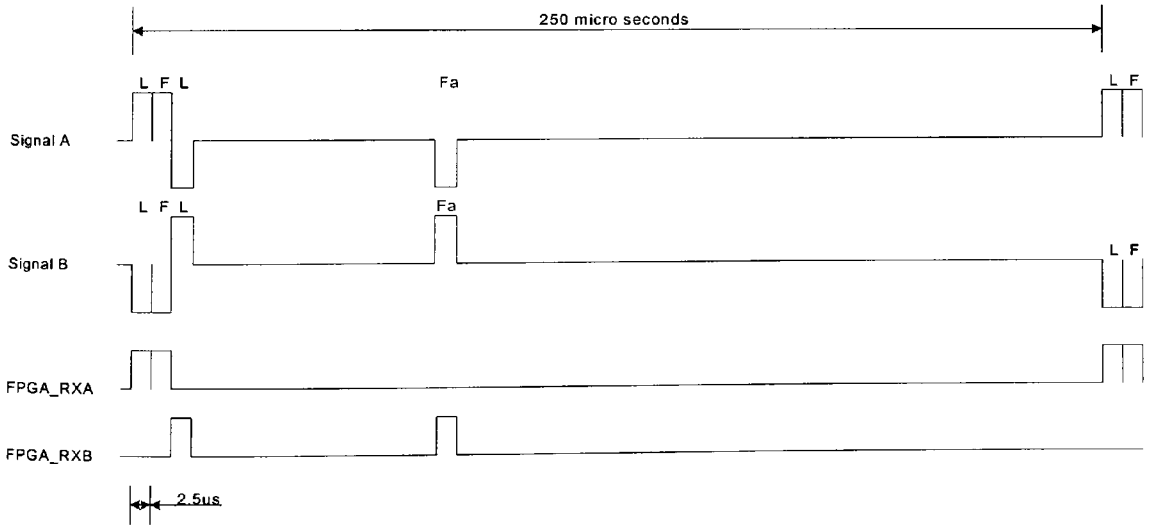


Figure 4 - Timing diagram for the analogue receiver circuit.

2.1 The reference voltage and threshold level circuit.

The schematic in figure 5 shows the reference voltage and threshold level circuit design. The op-amp U4B produces the reference voltage COMM_RX. The current source composed of transistors Q1 and Q2 drives a current value through resistor R22 in order to establish a voltage threshold level above the 2.5V reference point. This threshold voltage is buffered by the op-amp U4A

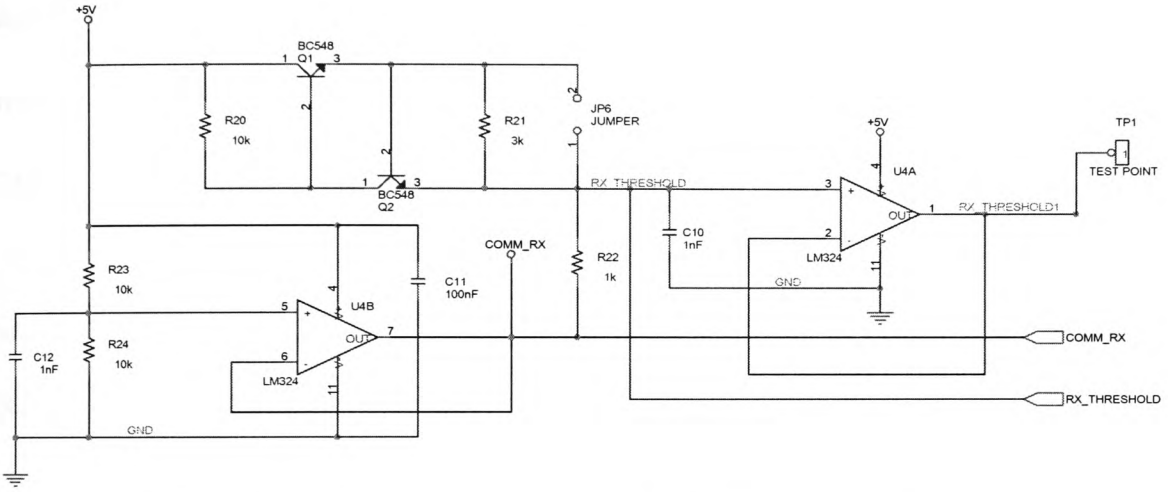


Figure 5 - The reference voltage signal threshold circuit.

The current source, shown in Figure 6 is composed of transistors Q1 and Q2 and resistors R20 and R21.

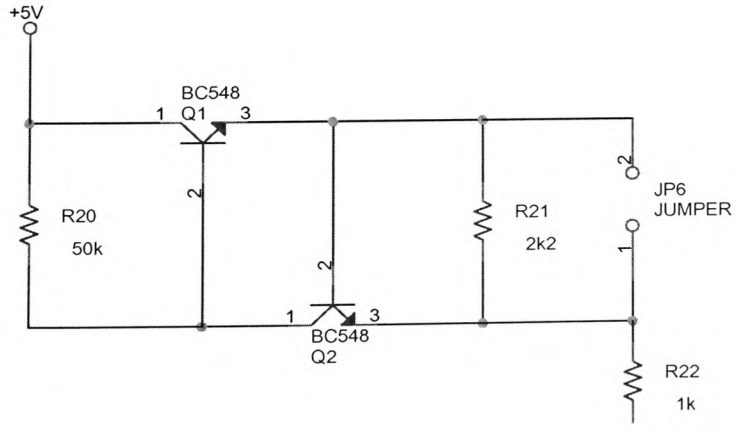


Figure 6 - The current source.

The current flowing into R22 (I_{R22}) is that flowing through R21 and R20. The equation for the circuit describing the current through these resistors is as follows.

$$I_{R22} = \frac{Q2_{Vbe}}{R21} + \left(\frac{5V - (Q2_{Vsat} + (2.5V + Vthres))}{R20} - Q1I_b \right)$$

The voltage dropped across the base emitter of Q2 is $Q2_{Vbe}$. The saturation voltage across Q2 is $Q2_{Vsat}$. The transistor base current for Q1 is $Q1I_b$. The choice of transistor will define $Q1I_b$, $Q2_{Vsat}$ and $Q2_{Vbe}$. The threshold voltage is chosen to be 250mV. The voltage at the emitter of Q2 will be 2.5V plus 250mV to give 2.75V. With these values derived from the BC548 transistor data-sheet, the resistors R20 and R21 can be selected to set the output current [3]. If R21 is chosen to be 3K and R20 is 50K then the output current is represented by equation 1.

$$250\mu A = \frac{0.5V}{2K2\Omega} + \left(\frac{5V - (1V + (2.5V + 250mV))}{50K\Omega} - 1.5\mu A \right) \quad \text{Equation 1.}$$

A value of 250μA through a 1k resistor induces a voltage drop of 250mV. The circuit accuracy will depend on the power supply tolerance and component tolerances, however bench tests have confirmed that the circuit operation is adequate for this application.

2.2 The buffer circuit.

The buffer circuit is shown in figure 7. The diodes D5 and D6 perform voltage clamping of signal A and signal B. Signal A and signal B are taken from the

transformer secondary winding. The capacitors C17 and C18 are not fitted in this application as isolation from the telephone line is achieved with the transformer. In this case zero ohm links have been fitted instead. The resistors R38 and R41 pull signal A and signal B to the 2.5V reference point COMM_RX. The LM319 comparators U6A and U6B compare signal A and signal B to the voltage threshold RX_THRESHOLD level, which sits 250mV above COMM_RX. The comparator output goes high when the comparator positive terminal exceeds the threshold voltage level at the negative terminal. The comparator output is pulled to the 3.3V FPGA supply voltage via resistors R36 and R39. The signals FPGA2_RXA and FPGA2_RXB are connected to the FPGA. These signals correspond to signals FPGA_RXA and FPGA_RXB in figure 4. Figure 8 and figure 9 show oscilloscope plots of the received AMI signal on channel 1 and the receiver output on channel 2. Figure 8 shows how the positive pulses of the AMI signal are extracted from the waveform. Channel 2 represents the buffered positives pulses that are sent to the FPGA. In figure 9, channel 2 shows how the negative pulses are extracted from the AMI signal.

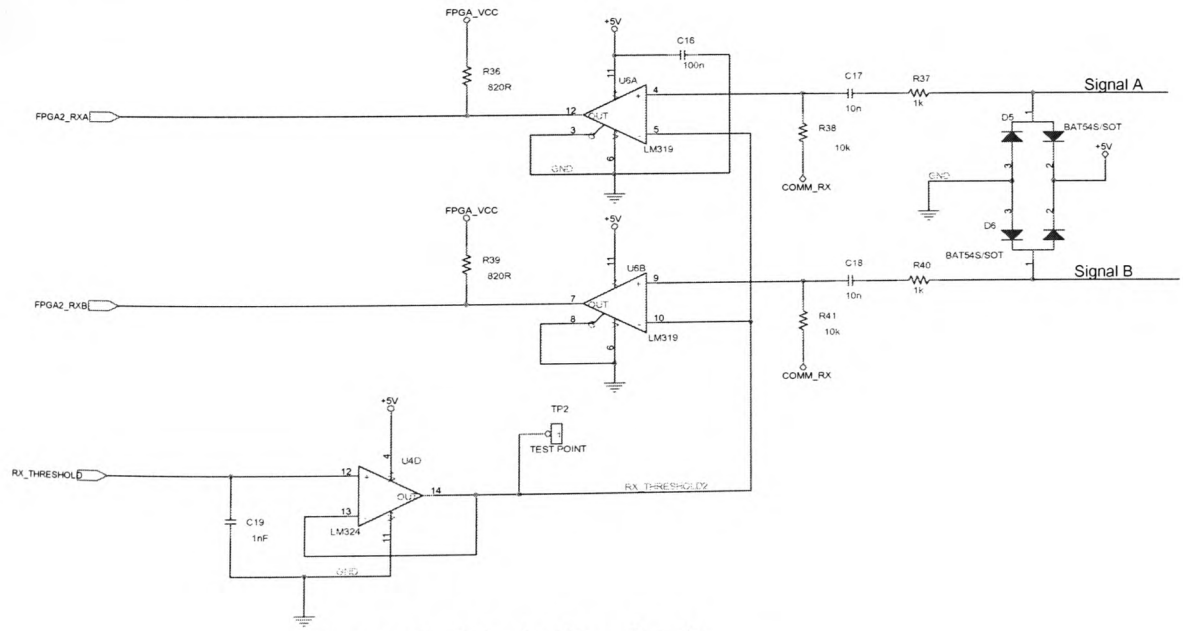


Figure 7 - The buffer circuit.

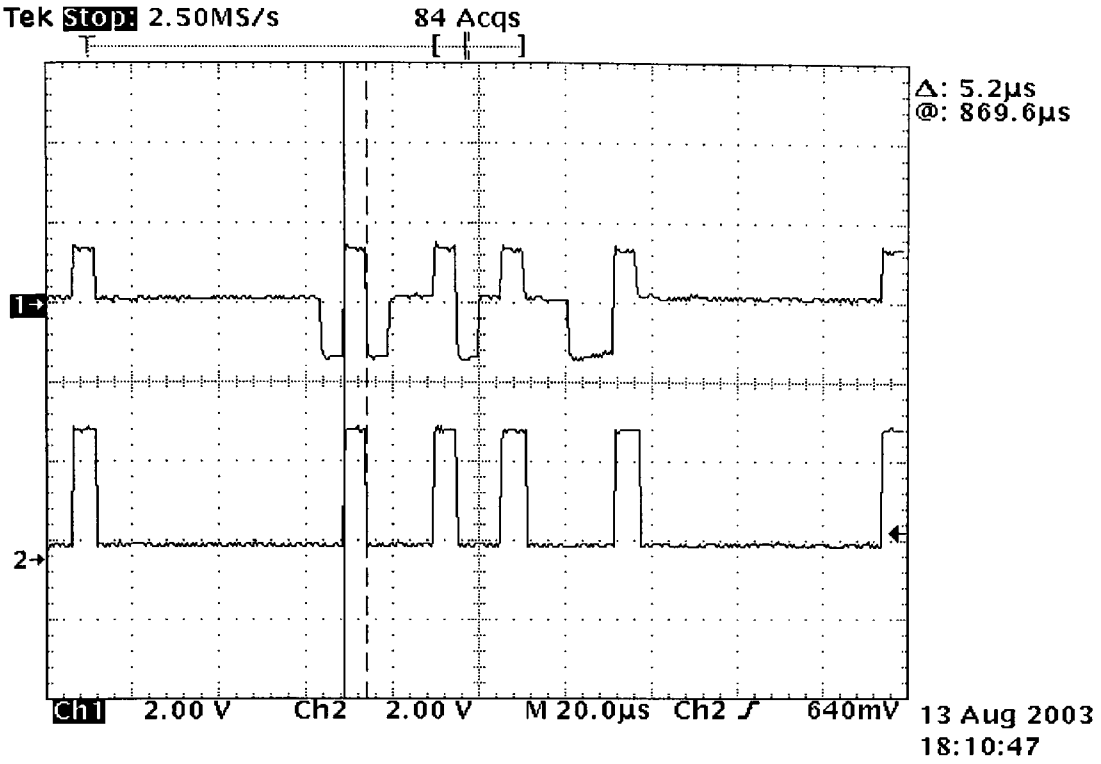


Figure 8 - The receiver circuit buffering the positive AMI pulses.

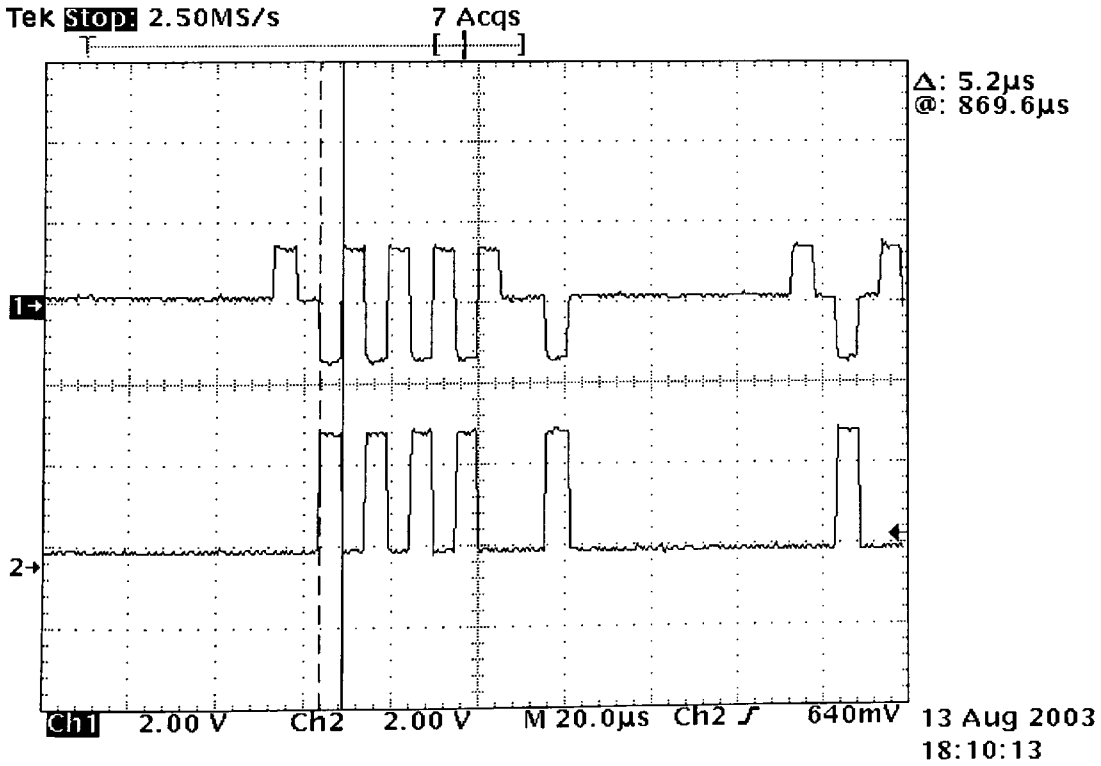


Figure 9 - The receiver circuit buffering the negative AMI pulses.

3 The Analogue signal transmitter.

The analogue signal transmitter is used to produce the differential signal required for data transmission on the S bus. The analogue signal transmitter is identical for both NT and TE designs. Recommendation I.430 defines signal transmission requirements for pulse shape, jitter requirements, termination impedance, balance about earth and output impedance. The development process will address the pulse shape and jitter requirements for the TE case. Traditionally design engineers have avoided the use of FPGAs due to a lack of skill in the area coupled with the complexity of the analogue design requirements for the signal receiver and transmitter circuits. The analogue signal transmitter must meet the requirements set out in recommendation I.430 before certification for use on the telephone lines can be obtained. The analogue circuits have been designed here to demonstrate that FPGAs have a realistic application in this area.

The analogue signal transmitter is isolated from the telephone line using a transformer. Figure 10 presents a block diagram of the differential driver. The system employs two amplifiers to create a push-pull network. The amplifiers interface directly to the transformer. The system operation can be demonstrated with reference to the timing diagram in figure 11. Figure 12 presents the amplifier required to produce signal A. A second amplifier using the same design configuration as amplifier A is employed to produce signal B. Only two signals are required from the FPGA and are applied to the amplifier circuits as shown in figure 10. This produces the differential signal. In the actual implementation the signals POS_PULSE and NEG_PULSE were inverted in the FPGA and these four

connections were used to drive the differential signal transmitter. This implementation uses two extra FPGA pins. In hindsight, two pins would have been sufficient and the signal inversion to the amplifier input terminals could be achieved on the PCB. The signal amplitude from the FPGA is 3V referenced to ground. The amplifier circuit is power by +/- 5V and thus its output will swing from +3V to –3V with the current arrangement. This signal is applied to a transformer with a turns-ratio of 2:1. The output of the transformer is connected to the telephone line.

Although further validation of the analogue design is required, the I.430 transmit pulse mask, shown in figure 13 has been complied with. The pulse mask in figure 13 defines the minimum and maximum voltage amplitude and period for each transmitted pulse. This objective of this test is to verify that the amplitude of the transmitted pulses are within the allowed limits. The pulse mask outlines the allowed variance that the pulse amplitude may deviate about the nominal pulse line (shown with the dashed line). The nominal transmitted pulse amplitude on the telephone as defined by I.430 is 750mV across 50Ω. I.430 states that the pulse amplitude may vary by 10% above and below this level to give an amplitude range of 675mV to 825mV. Figure 14 shows the transmitted waveform measured on the line across a 50Ω termination impedance. The pulse mask limitations applies to each positive and negative pulse. In this case the oscilloscope measurement cursors are set up to measure the amplitude of the positive pulses. The measured amplitude level was 700mV. A diode-clamping network may be added to reduce the amplitude overshoot evident in the oscilloscope plot. The FPGA circuit was configured to produce an INFO1 signal to facilitate pulse mask testing.

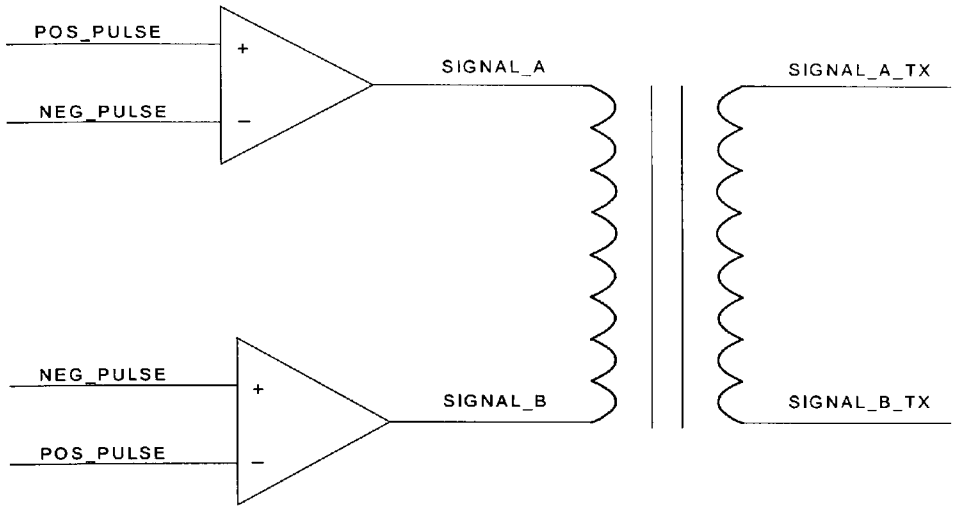


Figure 10 - Differential signal driver.

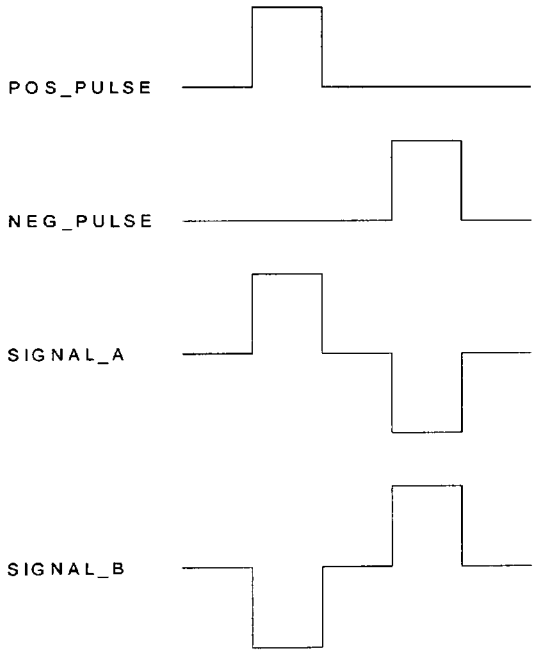


Figure 11 - Differential driver timing diagram.

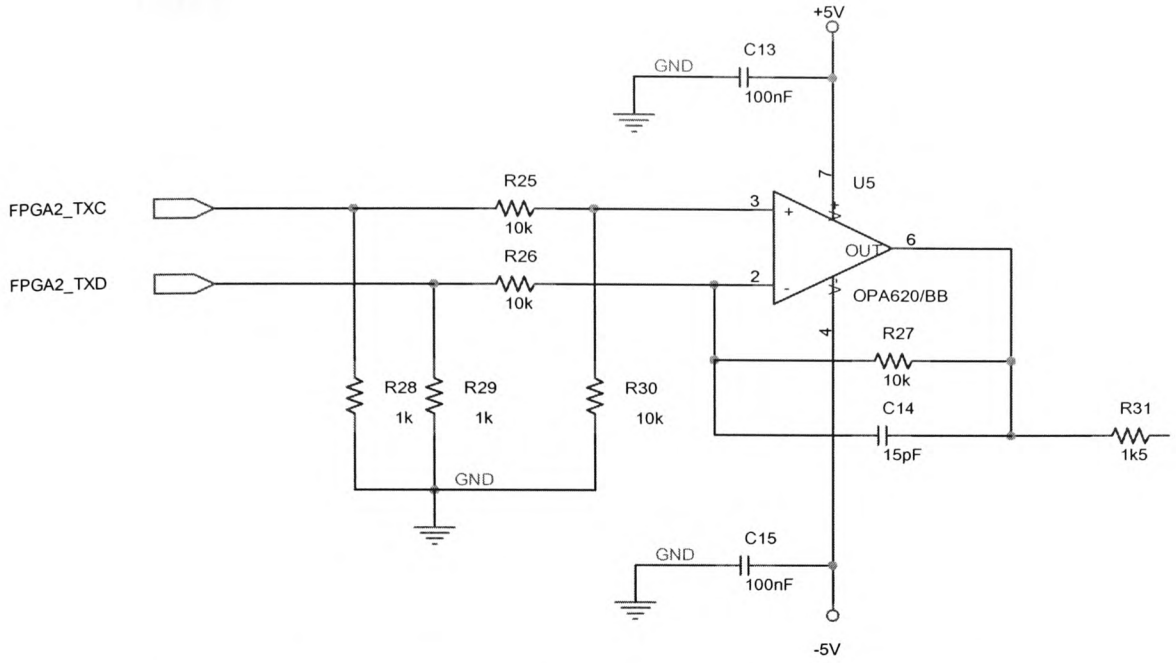


Figure 12 - The amplifier circuit used in the differential driver circuit.

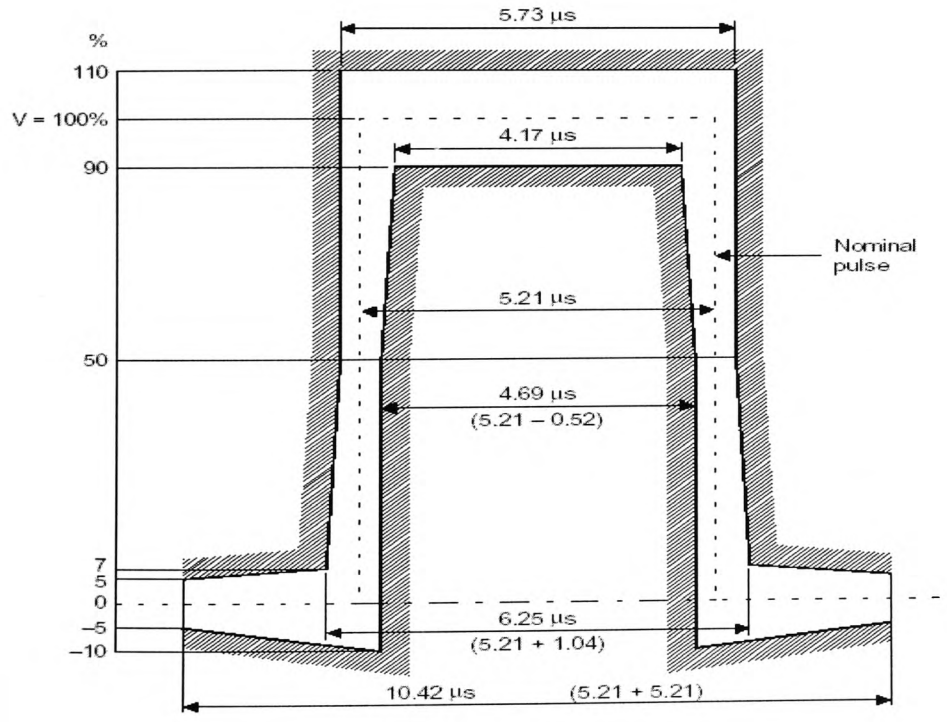


Figure 13 - The pulse mask for the signal transmitted by the TE [2].

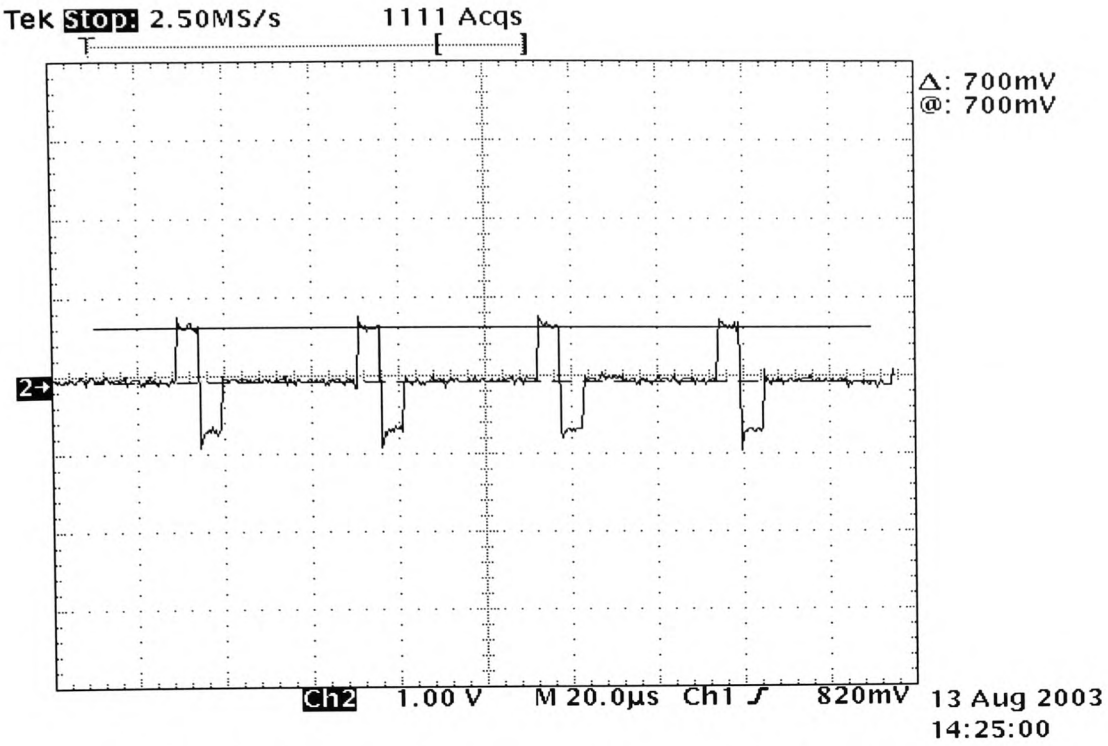


Figure 14 - The signal measured at the output of the TE.

4 The FPGA Firmware design for the Terminal equipment.

The FPGA firmware for the TE implements the S bus synchronisation, phase locked loop (PLL), maintenance channel interface, BERT test, and the signal coding for transmission. A block diagram for the TE firmware is shown in figure 15. Three functional blocks are shown. The SBUS_PLL block performs the synchronisation and clock phase locking with the S bus signal from the NT. The sampling of the S bus AMI signal occurs synchronously with the master clock. The master clock labeled MCLK_RX on the block diagram runs at a frequency of 192kHz and is derived from the 15.36MHz reference crystal. The synchronisation process begins with a reset to the FPGA. This initialises the synchronisation circuitry and the next valid symbol on the S bus resets the master clock generator. This action ensures that the next positive going edge of the master clock should coincide roughly with the middle of each subsequent S bus symbol. The master clock should coincide roughly with the center of each data symbol over the period of time the synchronisation process requires.

The bit rate on the S bus is 192kHz and the symbol period is 5.2μs. The stability of the master clock is defined by the stability of the 15.36MHz reference crystal oscillator. Each symbol has a period of 5.2μs and a frame lasts for 250μs. Three valid frames are required to allow synchronisation. The stability of the reference crystal should ensure that the edge of the master clock does not move by more than 2.6μs over a time period of three data frames or 750μs. The required stability of the master clock is defined by I.430 as 100 parts per million (ppm). In practice engineers employ crystals with a stability of 50ppm to allow for a drift in crystal manufacturing tolerance. In this application a reference oscillator with a

tolerance of 100ppm is used. With a 100ppm tolerance the edge of the master clock is expected to drift by 25ns per frame (100ppm/1000000 x 250μs = 25ns) or 750ns over a period of three frames. Once synchronisation is achieved the signal SYNC goes active and the PLL is enabled. The PLL is implemented with digital components only and is hence referred to as an all-digital phase locked loop (ADPLL). The ADPLL performs fine-tuning of the clock phase with respect to the center point of each symbol. As explained in the introduction, the timing information is incorporated within the data stream through coding rule violations that occur at defined intervals. These coding violations can be used to facilitate synchronisation with the S bus.

The LOOPBACK block performs synchronisation with the maintenance channel and requests a loopback on a B channel via the NT. The Q_BITS signal represents the data to be sent in the Q channel, which is the maintenance channel in the TE to NT direction of transmission. The Q channel carries the loopback request to the NT. The request is sent continuously until the NT responds by activating the Loopback and returning a request acknowledge. The TE sends the request continuously for the duration of a test. The NT clears the loopback when it no longer receives the loopback request from the TE. Once the NT enables a loopback the PRBS generator transmits a PRBS sequence toward the NT. The PRBS sequence is returned to the TE and a PRBS receiver in the TE monitors the returned sequence for errors. Errors that occur are counted and the result is available on the PRBS error count output from the block.

The INFOGEN_TENT block generates the AMI signal according to the coding rules defined by recommendation I.430 and performs the S bus framing of data from the TE to the NT. The B channel data and Q channel data are encoded and

inserted into the data frame. On the block diagram the B1_DATA and the B2_DATA are inserted into the B1 channel and B2 channel. As already explained in the analogue transmission section two amplifier circuits are employed to produce the AMI signal. Each amplifier requires two control signals. The signals POSA_PULSE, NEGB_PULSE, POSC_PULSE and NEGD_PULSE are the control signals for the analogue transmission circuit.

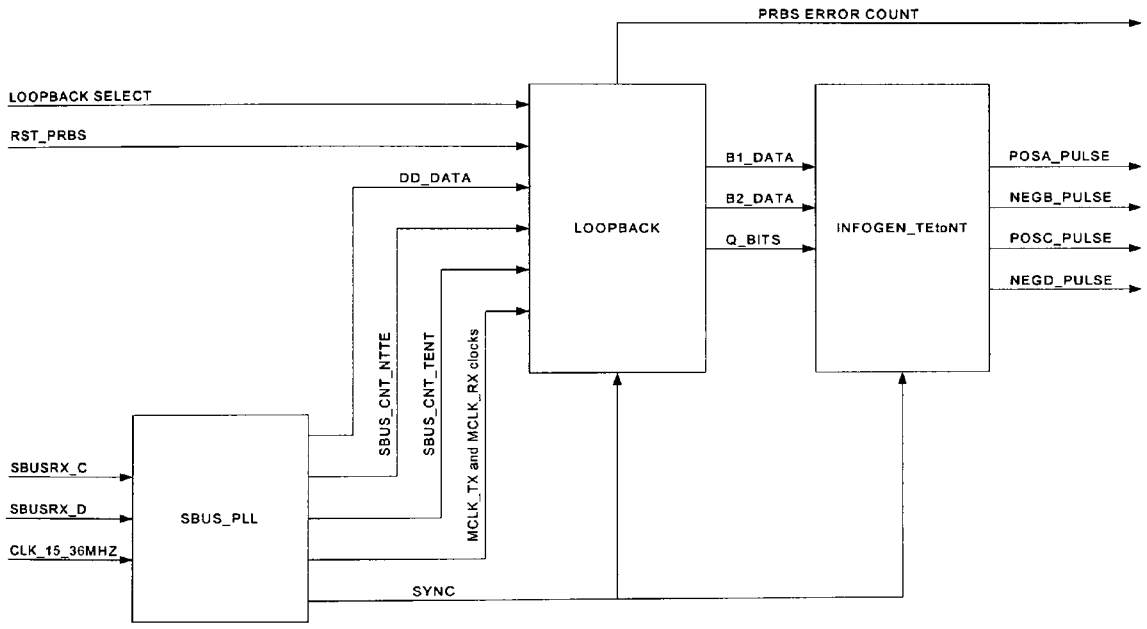


Figure 15 - The TE firmware block diagram.

4.1 The S bus synchronisation and ADPLL function.

This section describes the operation of the SBUS_PLL block. The SBUS_PLL performs two distinct functions, synchronisation and phase locking with the ISDN S bus. Although both functions are interlinked the synchronisation process will be described firstly. The ADPLL is a sub-component of the system. The phase locking process is enabled when synchronisation with the S bus is complete. The synchronisation process involves sampling each symbol and hunting for permitted violations that comply with the coding rules. Once three sets of consecutive violations have been found the synchronisation process is considered complete. Alternatively, once synchronised, if three sets of consecutive violations are not found, then synchronisation can be regarded as lost. This method has been adopted to design a level of noise tolerance into the system operation.

The timing information is embedded within the data frame from the NT and only when synchronisation is complete can the position of individual symbols within a frame be identified. The synchronisation process starts by resetting the sampling clock to coincide roughly with the center point of each symbol on the S bus. The term roughly is used here as the sampling clock is not locked but free running with its stability defined by the reference crystal oscillator. The sampling clock signal will be labeled as MCLK_RX and figure 16 presents a timing diagram that shows its relationship with symbols on the S bus. The sampling clock MCLK_RX clocks the state of each S bus symbol into the FPGA on the rising edge to produce the signal labeled synchronised AMI data.

Figure 17 shows two frames of data. The data frame detail has been reduced to focus on the coding rule violations within the data frames. Each frame lasts for

250μS. In the timing diagram, the first L symbol and the following F symbol are the same polarity contrary to coding rules and this creates the first violation. The second L symbol and the first B1 symbol have the same polarity and this constitutes the second violation. Once the first violation has been identified, the second violation must occur within thirteen symbol positions from the F symbol position (14-bit criterion). If the NT cannot create a violation before the Fa symbol position then the Fa symbol will be used to set the second violation. This statement provides the basis for the synchronisation system specification.

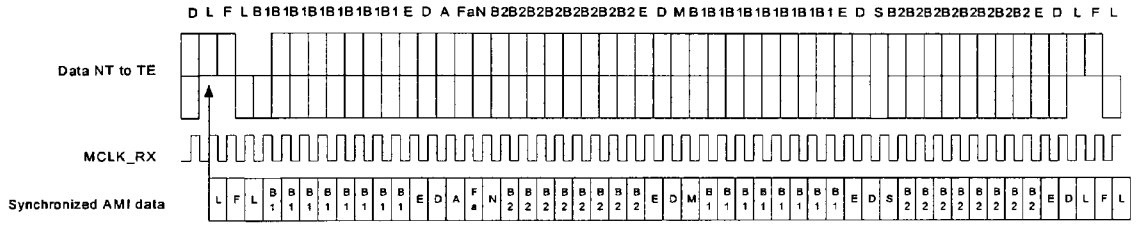


Figure 16 - S bus data synchronized to the MCLK_RX

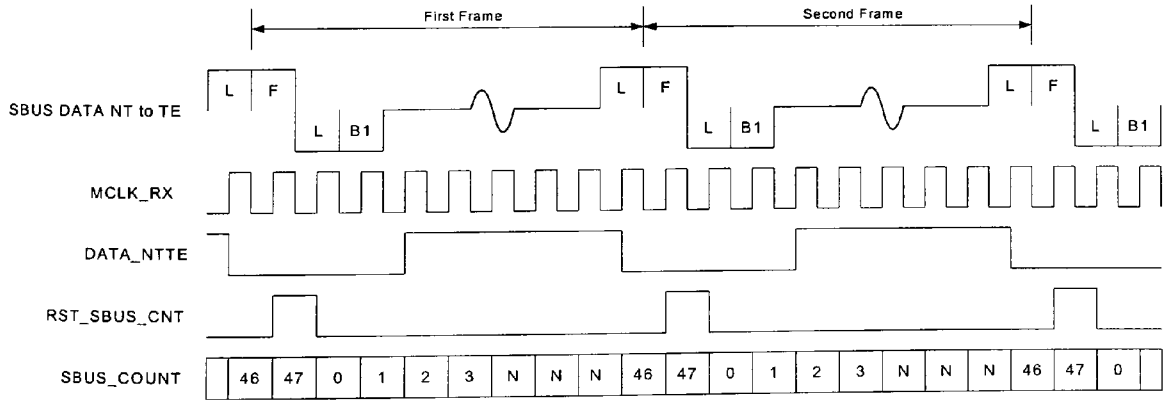


Figure 17 - Timing diagram showing two frames of S bus data.

The specification for the synchronisation circuit deals primarily with the hunt for three consecutive sets of coding violations. The block diagram for the system is presented in figure 1, of appendix A. The block diagram is composed of architectural blocks and a state machine controller. The architectural blocks are FIX_POLARITY_ARCH, SBUS_FLAGS_ARCH, CRIT_CNT_ARCH, SBUS_CNT_ARCH, INFO0_WDOG_ARCH and DLL. These facilitate the synchronisation process carried out by the state machine controller, SBUSDLL_CON. The state machine controller performs all the decision making for the system based on the information from the architectural blocks. The following section describes the operation of each architectural block and the algorithm implemented by the state machine is presented.

The FIX_POLARITY_ARCH block.

The FIX_POLARITY_ARCH block is employed to facilitate the search for the permitted coding rule violations. It performs signal multiplexing and signal synchronisation with the sampling clock. The signal inputs and outputs are defined as follows.

The FIX_POLARITY_ARCH block inputs and outputs.

SBUSRX_A: This is an input signal from the analogue signal receiver.

SBUSRX_B: This is an input signal from the analogue signal receiver.

SET_POS_PULSE: This is a control input that routes the signal SBUSRX_A to the output POS_PULSE and routes the signal SBUSRX_B to the output NEG_PULSE.

SET_NEG_PULSE: This is a control input that routes the signal SBUSRX_A to the output NEG_PULSE and routes the signal SBUSRX_B to the output POS_PULSE.

RST: This synchronously resets the system.

CLK: This is the clock signal for the system and is derived from the phase locked loop (DLL block). The clock samples the data on the line represented by the input signals SBUSRX_A and SBUSRX_B. The clock frequency is 192kHz.

C15_36MHZ: This input is from the reference crystal and has a frequency of 15.36MHz.

The output signals from the FIX_POLARITY_ARCH block are defined as follows.

SBUS_DD: This output signal presents the AMI data extracted from the S bus in binary form.

PULSE_ON_LINE: This output signal is produced from the logical OR of the input signals SBUSRX_A and SBUSRX_B and synchronized to the 15.36MHz clock.

PULSE: This output signal is produced from the logical OR of the input signals SBUSRX_A and SBUSRX_B and synchronised to the clock signal CLK (192kHz).

POS_PULSE: The signal SBUSRX_A or SBUSRX_B is routed through to POS_PULSE depending on the logical state of the signals SET_POS_PULSE and SET_NEG_PULSE. The signal POS_PULSE is connected to the state machine.

NEG_PULSE: The signal SBUSRX_A or SBUSRX_B is routed through to NEG_PULSE depending on the logical state of the signals SET_POS_PULSE and SET_NEG_PULSE. The signal NEG_PULSE is connected to the state machine.

FL_PAIR: This signal identifies the FL symbol pair on the S bus. When synchronization has been achieved the signal FLPAIR produces the reference signal used by the phase locked loop to lock the receiver and transmit clocks.

The FIX_POLARITY_ARCH operation.

The schematic diagram representing the operation of the FIX_POLARITY_ARCH block is shown in figure 18. VHDL was employed to produce the actual FPGA implementation. Before violations can be successfully identified the polarity of the signal on the S bus must be determined. As already described in the analogue signal receiver section, two digital signals representing data on the telephone line are received by the FPGA firmware. These input signals are labeled here as SBUSRX_A and SBUSRX_B. SBUSRX_A represents positive AMI pulses on the S bus and SBUSRX_B represents negative AMI pulses on the S bus. As the polarity of the signal on the S bus is undefined, one of the two cases presented in figure 19 could exist. Figure 19 presents two sets of synchronisation pulses with different signal polarities. Due to the way the analogue signal receiver is implemented the first violation may appear on SBUSRX_A or SBUSRX_B. The synchronisation circuit is required to determine which case exists before synchronisation can be achieved.

The reason for this requirement is explained as follows. If the synchronisation process were to assume that case A was true, then scanning SBUSRX_A would correctly identify the first violation. Once the violation is identified SBUSRX_A is scanned and the second violation should be identified within fourteen symbol periods (5.2us x 14 = 72.8μs). However, if case b exists the second violation would be perceived as the first and a second violation would not be seen within fourteen

symbols from this point. Synchronisation would not be possible in this case and some means of switching the scanning process to deliberately search for the first violation on SBUSRX_B rather than SBUSRX_A is required. This is achieved with the arrangement shown in figure 18. The input signals SBUSRX_A and SBUSRX_B are routed to the output signals POS_PULSE and NEG_PULSE based on the state of SET_POS_POL and SET_NEG_POL. The truth table for the multiplexer operation is shown in table 1.

SET_POS_POL	SET_NEG_POL	POS_PULSE	NEG_PULSE
LOGIC 1	LOGIC 0	SBUSRX_A	SBUSRX_B
LOGIC 0	LOGIC 1	SBUSRX_B	SBUSRX_A

Table 1 - The FIX_POLARITY_ARCH multiplexer operation.

This allows the state machine to hunt for the correct sequence of coding rule violations independently of the signal polarity on the telephone line.

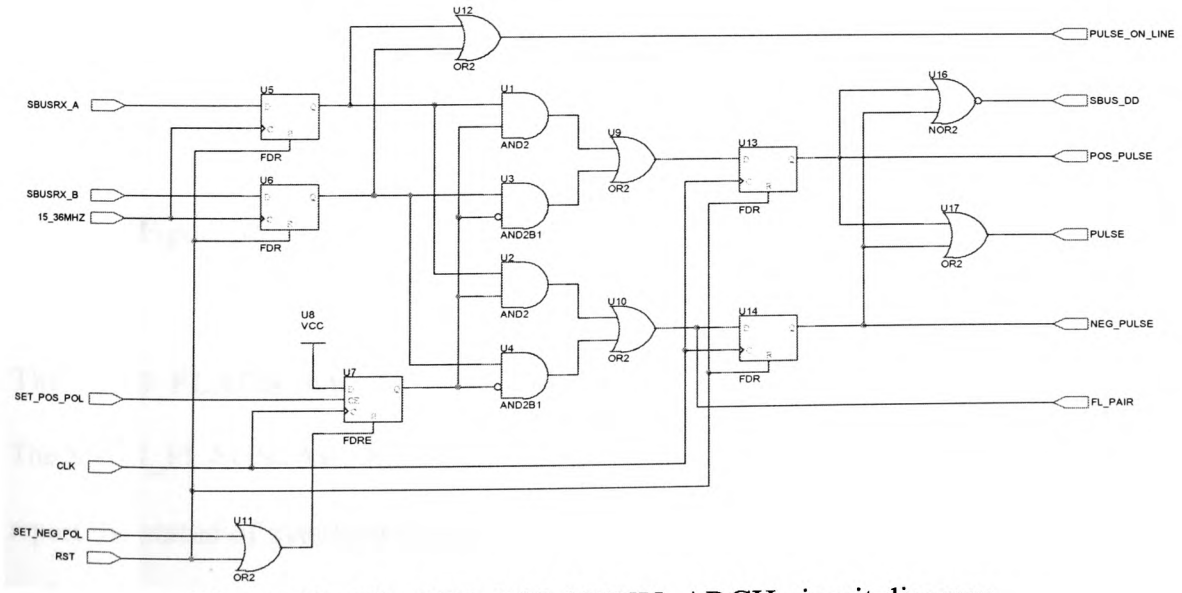


Figure 18 - The FIX_POLARITY_ARCH circuit diagram.

The signal SBUS_DD represents the AMI signal in binary. The NOR result of the signals POS_PULSE and NEG_PULSE ensures that positive or negative pulses are represented by a binary zero and the absence of pulses are represented by a logic zero. The FL pair is monitored when synchronisation with the S bus is established. The signal FL_PAIR represents the state of the L symbol following the F symbol on the line. Once synchronisation is achieved the signal FL_PAIR represents the only timing event that is guaranteed within a data frame. The phase locked loop can then achieve clock edge alignment using this signal as the clock edge reference.

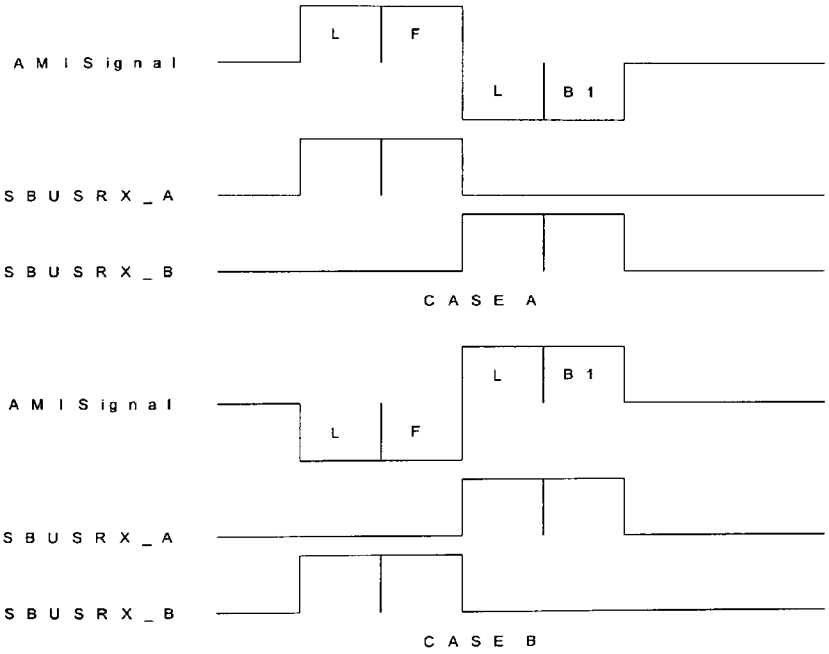


Figure 19 - Diagram showing AMI signals with different polarity.

The SBUS_FLAGS_ARCH.

The SBUS_FLAGS_ARCH implements three status flags for the system. The flags report the status of synchronization with the NT, when there is no signal on the line

(INFO0 condition) and an indication of signal polarity on the S busThis blocks input and output signals are defined as follows.

The SBUS_FLAGS_ARCH input signals.

CLK: This is the clock signal driven by MCLK_RX.

CLR_SYNC: This clears the sync flag to logic 0. The operation occurs synchronously with the positive edge of CLK.

SET_SYNC: This sets the sync flag to logic 1. The operation occurs synchronously with the positive edge of CLK.

CLR_INFO0: This clears the INFO0 flag to logic 0. The operation occurs synchronously with the positive edge of CLK.

SET_INFO0: This sets the INFO0 flag to logic 1. The operation occurs synchronously with the positive edge of CLK.

CLR_POLARITY: This clears the POLARITY flag to logic 0. The operation occurs synchronously with the positive edge of CLK.

SET_POLARITY: This sets the POLARITY flag to logic 0. The operation occurs synchronously with the positive edge of CLK.

The SBUS_FLAGS_ARCH output signals.

SYNC: This is the Synchronisation flag and when set to logic 1 indicates that synchronisation has been achieved.

INFO0: This is the INFO 0 signal condition flag and when set to logic 1 indicates that there is no signal on the line.

POLARITY: This indicates the signal polarity on the line. When set to logic 0, this indicates negative polarity i.e. case A as in figure 19 is indicated and when set to logic 1 positive polarity is indicated.

The CRIT_CNT_ARCH block.

The CRIT_CNT_ARCH block implements two synchronous counters, counter 1 and counter 2. Each counter has with four states each. One of the counters is advanced on successfully identifying a set of coding rule violations. The other counter is advanced each time a set of violations cannot be found. The counters are advanced and reset by the state-machine during the process. The inputs and outputs are described as follows.

The CRIT_CNT_ARCH inputs.

CLK: This is the clock signal driven by MCLK_RX.

RST_CCNT1: This signal is used to reset counter 1 to 0.

ADV_CCNT1: This signal is used to advance counter 1 through states 0 to 3.

RST_CCNT2: This signal is used to reset counter 2 to 0.

ADV_CCNT2: This signal is used to advance counter 2 through states 0 to 3.

The CRIT_CNT_ARCH outputs.

CCNT1: This signal goes active when the counter state is 3.

CCNT2: This signal goes active when the counter state is 3.

The SBUS_CNT_ARCH block.

The SBUS_CNT_ARCH block implements two synchronous counters with 48 states each. A data frame on the S bus contains 48 bits of data. One counter is used to identify each data bit on the S bus for the NT to TE direction of transmission. The other counter is used to identify data bits in the TE to NT direction of transmission. The counter output is broadcast to the other hardware blocks of the firmware in order to facilitate synchronisation of events. Once synchronisation is achieved the counter outputs can be decoded to indicate when data receive and transmit operations should take place. The timing diagram for the counter is shown in figure 20. The NT to TE data frame is shown. The sampling clock labeled as RX_CLK clocks the data from the S bus into the TE to produce the synchronised data stream, SBUS_NTTE_SYNCED as shown. The state machine generates the signal RST_BITCNT_NTTE when the first coding rule violation is identified. The first violation should always be identified after the F symbol is sampled. The counter is consistently reset at this point in the frame.

The second violation should then occur before the counter reaches the fourteenth state, where the Fa or the N symbol will force a violation. The counter is decoded to produce the signal TV2_UP when the count reaches 13. The signal TV2_UP provides a time-out signal for the state machine during its hunt for the second violation. There is a two-bit offset between data frames in the NT to TE direction and the TE to NT direction. Once synchronisation to the NT to TE direction is achieved all the necessary information to derive the TE to NT data frames is known. The inverse of the sampling clock will produce the transmit clock labeled as TX_CLK. The signal RST_BITCNT_NTTE is synchronised to the TX_CLK to produce RST_BITCNT_NTTE_Q. The signal

RST_BITCNT_NTTE_Q can then be delayed by one more clock period of TX_CLK to produce the signal RST_BITCNT_TENT. The signal RST_BITCNT_TENT can now be used to reset the second 48 bit counter and the symbols in the TE to NT direction can be identified.

It was decided at the design stage to develop the solution so that the count sequence representing data in the NT to TE direction is the same as that in the TE to NT direction. Using this design constraint the F symbol in both the NT to TE direction and the TE to NT direction would be identified by state 47. This reduces confusion when visualizing the relationship between both directions of data transmission and errors due to bad decoding can be avoided.

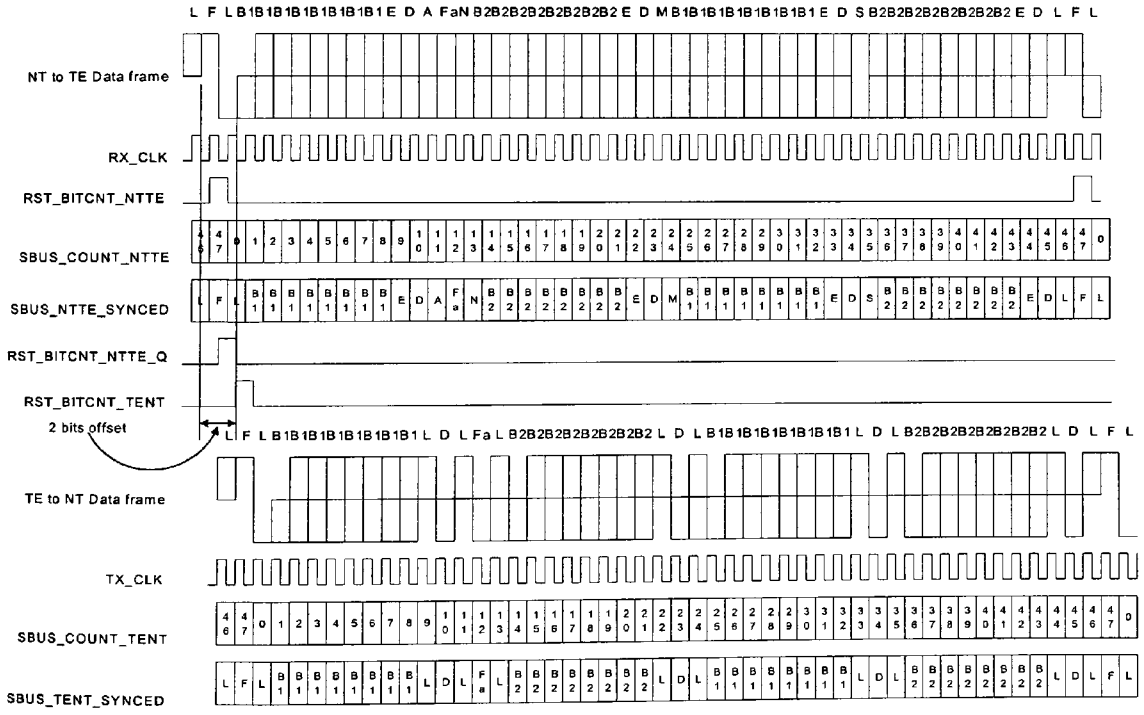


Figure 20 - The timing diagram for the SBUS_CNT_ARCH block.

The SBUS_CNT_ARCH block inputs.

RST_BITCNT_NTTE: This resets the counter.

CLK_TX: This is the transmit clock signal used to clock data out of the TE.

CLK_RX: This is the sampling clock signal used to clock data into the TE.

The SBUS_CNT_ARCH block outputs.

SBUS_TENT: This is the count sequence identifying the individual symbols in the TE to NT direction of transmission. The signal is a 6-bit bus.

SBUS_NTTE: This is the count sequence identifying the individual symbols in the NT to TE direction of transmission. The signal is a 6-bit bus.

TV2_UP: This signal goes high 72.8μs (14 bit positions) after the first violation is detected. This signal occurs in synchronism with RX_CLK when the count sequence reads 13.

FBIT: This signal occurs once every 250μs and is synchronised to RX_CLK. FBIT goes active when the count sequence reads 47. This signal provides a means of marking the number of frames passed over time.

The INFO0_WDOG_ARCH block.

The INFO0_WDOG_ARCH block is a watchdog timer that counts the number of frames that pass over time. The output TFRAME_UP will go high after 2ms or 8 frames in the absence of a reset on the signal RST_TFRAME. On successfully identifying a set of violations the signal RST_TFRAME is set high and the timer is reset.

The INFO0_WDOG_ARCH input signals.

RST_TFRAME: This signal comes from the controller and resets the watchdog timer.

FBIT: This signal comes from the SBUS_CNT_ARCH and advances the watchdog count every 250μs.

CLK: This is connected to the RX_CLK signal.

The INFO0_WDOG_ARCH output signal.

TFRAME_UP: If the watchdog timer is not reset, this signal goes active after 8 frames.

The SBUSDLL_CON block.

The SBUSDLL_CON block is the state machine controller for the synchronisation process. The state machine has eight states. The operation of the controller is specified in the algorithmic state machine (ASM) chart in figure 22a, 22b, and 22c. The state machine makes all the decisions for the synchronisation based on the output signals from the architectural blocks already described. The objective of the state machine is to hunt for three consecutive coding rule violations as defined by recommendation I.430. Once three violations are identified the synchronisation flag is set. The following describes the function of each state along with the associated input and output signals. The state machine is clocked with the clock signal MCLK_RX. The state machine is reset by the signal RST. This is the hardware reset for the entire FPGA. In this implementation logic 0 is false and logic 1 is true. Figure 21 shows the timing diagram for the synchronisation process. Reference will

be made to this diagram during the discussion. The state sequence repeats indefinitely maintaining an indication of the current synchronisation status.

The description of the states in the SBUSDLL_CON state machine.

State S0: S0 is the default state and when in this state the flags in the SBUS_FLAGS_ARCH block are cleared. The watchdog timer is cleared. The polarity is set to positive polarity and SBUS_CNT_ARCH block is reset. The state sequence moves to S1 on the next cycle of the clock.

State S1: S1 is a wait state where the state sequence will remain until a pulse is sensed on the line. The signal PULSE is an output from the FIX_POLARITY_ARCH. If there is activity on the line this signal will go active. The state machine will remain in this state until a signal is received from the NT. If no signal exists on the line then the state machine will leave S1 only on a reset.

State S2: State S2 lasts for one clock cycle. In this state the watchdog timer is reset current polarity is checked. The polarity of the system is determined by firstly assuming that the polarity is positive. In other words it is assumed that the first violation appears on the signal POS_PULSE from FIX_POLARITY_ARCH. The signal POLARITY from the SBUS_FLAGS_ARCH block is read and as it was reset in S0 it will read false. This results in the output signal SET_POS_POL being asserted true for one clock cycle before moving to S3. The signal SET_POS_POL is sent to the FIX_POLARITY_ARCH and the truth table defined in table 1 applies. When the state machine cannot synchronise within eight frames then S2 is entered

and the polarity is changed. This is achieved by asserting SET_NEG_POL to allow hunting for violations on a negative polarity signal. The state machine may switch between different signal polarities a number of times before finally synchronising.

State S3: State S3 is a wait state. S3 will wait for a positive pulse on the line. The signal P_PULSE is received from the FIX_POLARITY_ARCH. When a positive pulse is identified the state sequence moves to S4.

State S4: State S4 hunts for another positive pulse following the one identified in S3. The signals N_PULSE and P_PULSE are received from the FIX_POLARITY_ARCH. If a negative pulse is identified when in S4 then the state sequence moves back to S3 and the hunt continues. When two positive pulses are found between which no negative pulse exists then a coding rule violation is identified. This can be seen in figure 21. The signal RST_SBUS_CNT resets the SBUS_CNT_ARCH block and the state sequence moves to S5.

State S5: When in S5 it is expected that logic 1 appears on the signal N_PULSE. At this point the N_PULSE should represent the L symbol following the F symbol. These two symbols are referred to as the FL pair. When N_PULSE is false it can be deduced that the second violation has not been detected and the state sequence returns to S3 to continue hunting. The state sequence will advance through the states S3 to S5 continually until either TFRAME_UP goes active forcing the input signal polarity to change of a valid FL pair is detected.

State S6: When a valid FL pair has been detected the state sequence moves to S6. The next signal that is detected should be logic 1 on N_PULSE only. This signal should occur before TV2_UP goes active to signal the 14-bit criterion time-out. When the second negative pulse is detected as shown in figure 21, the second violation is detected. This action advances the counter CCNT1 in the CRIT_CNT_ARCH block. If the time-out signal goes active or a pulse is detected on P_PULSE then the state sequence returns to S3. In this case the counter CCNT2 in the CRIT_CNT_ARCH block is advanced.

State S7: In state S7 the counters in the CRIT_CNT_ARCH block are checked. If CCNT1 is active then the SYNC flag is set to indicate synchronisation. If CCNT2 is active then the SYNC flag is reset to indicate loss of synchronisation.

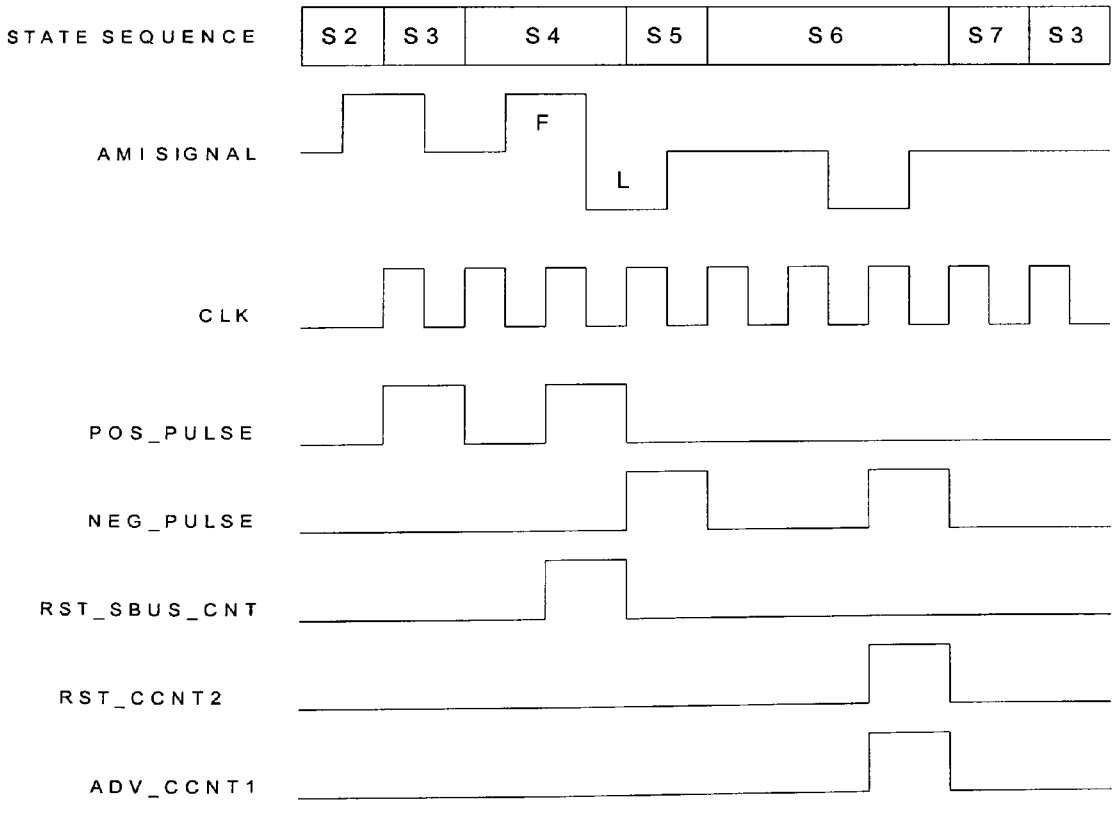


Figure 21 - The SBUSDLL_CON blocks timing diagram.

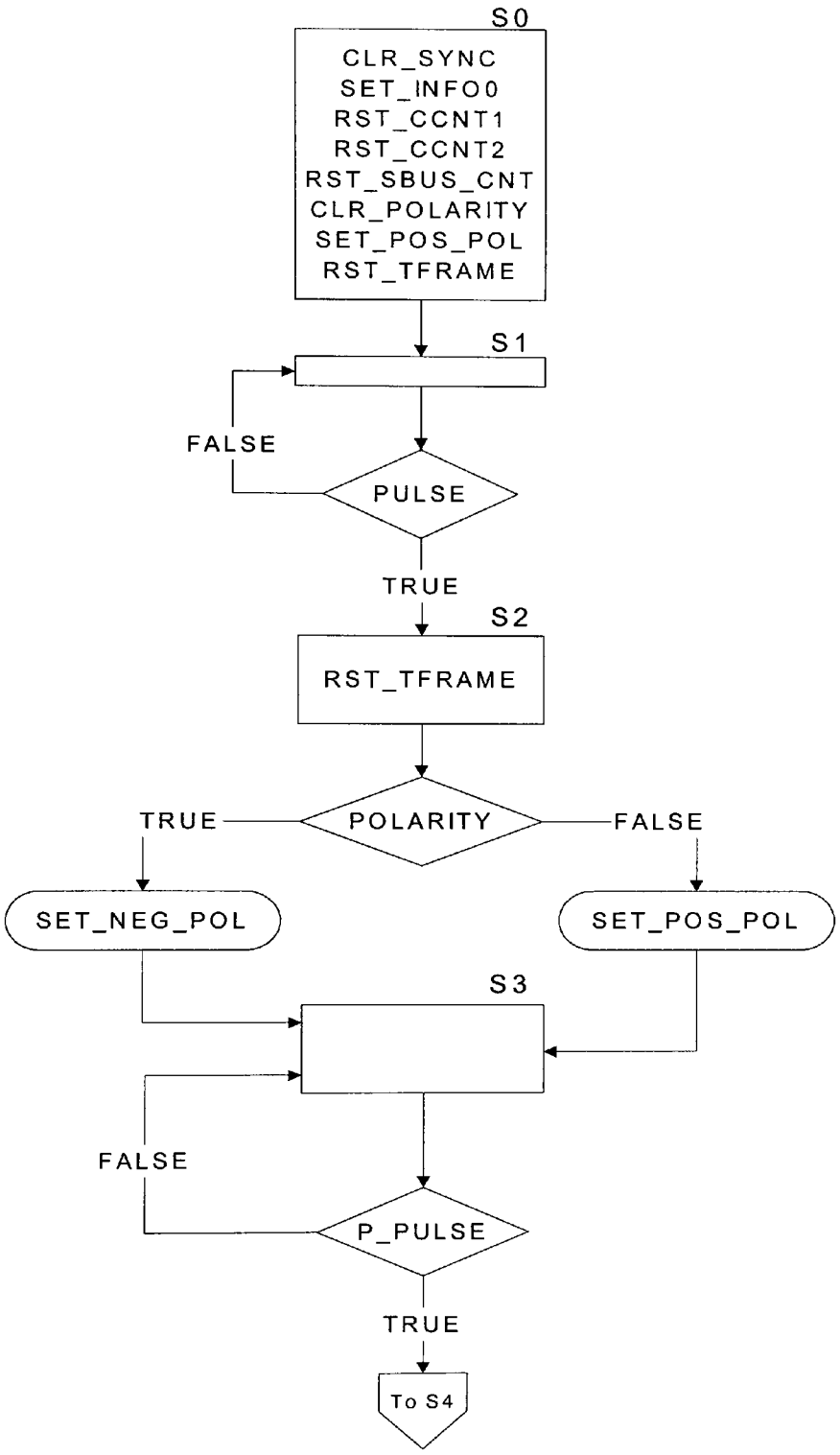


Figure22a - The SBUSDLL_CON state machine.

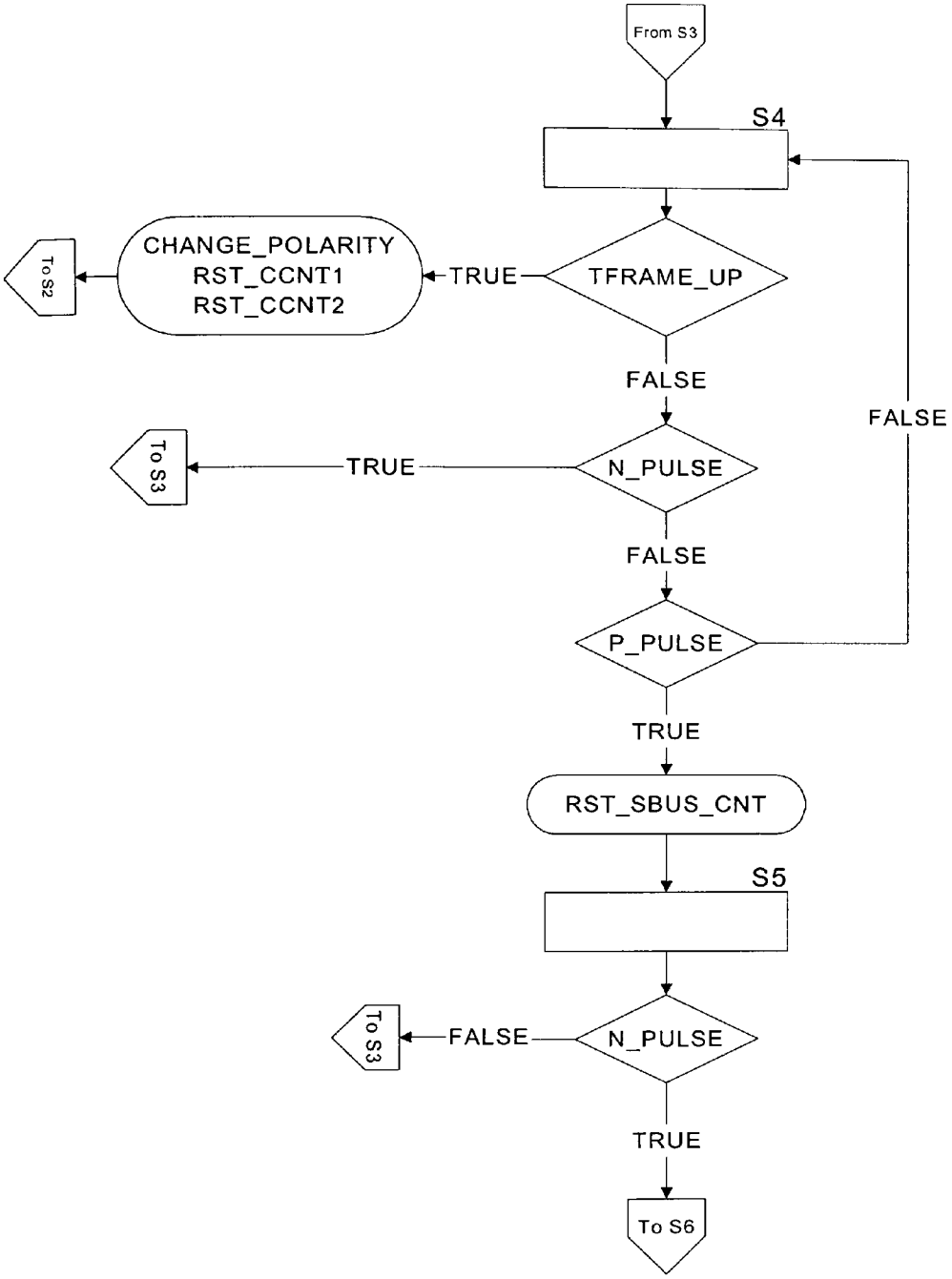


Figure22b - The SBUSDLL_CON state machine.

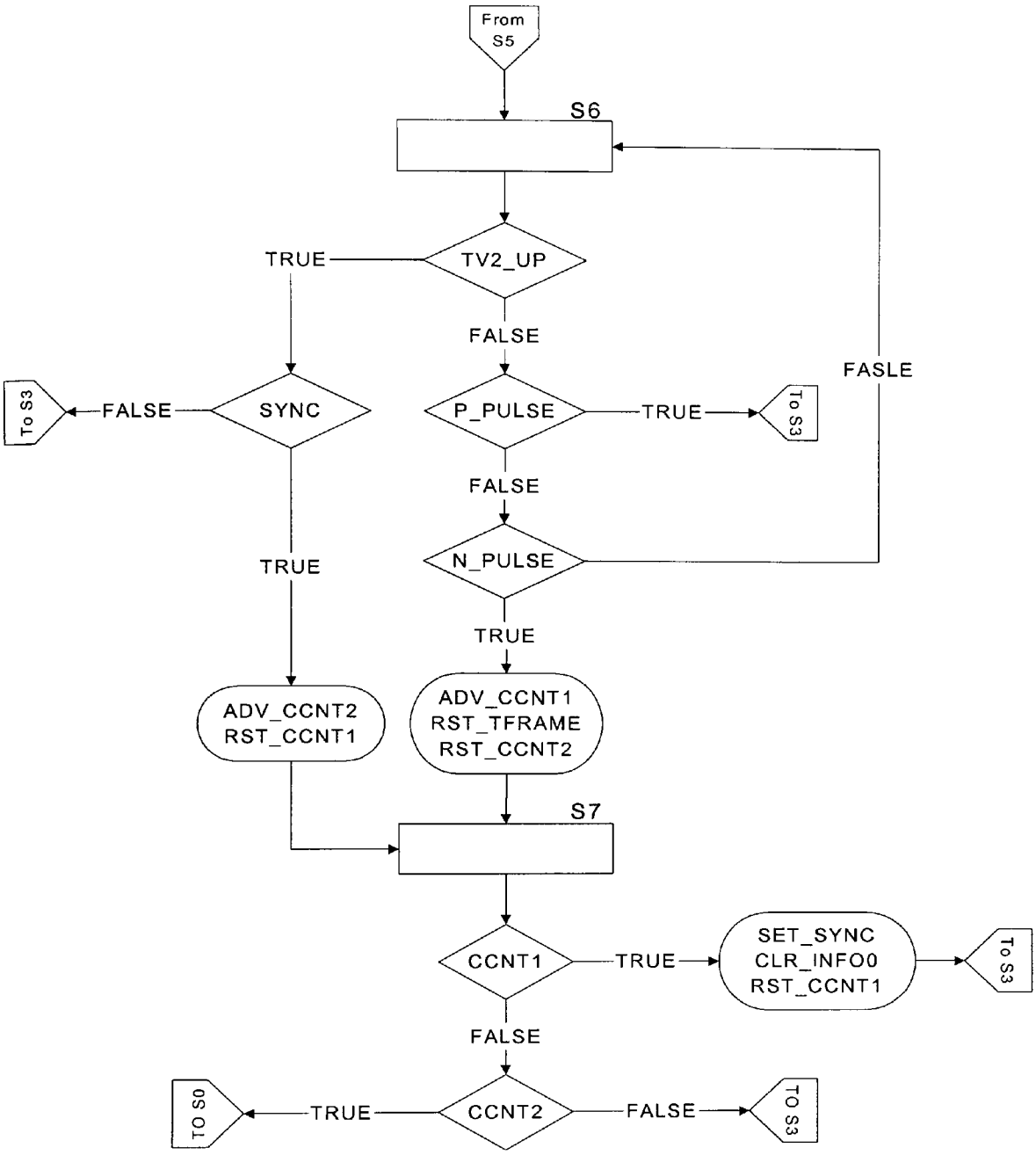


Figure22c - The SBUSDLL_CON state machine.

4.2 The all digital phase locked loop block.

The ADPLL is activated when synchronisation is achieved. Once synchronisation is achieved with the S bus the position of the FL pair is known. The FL pair is used as the timing reference point. The signal RST_SBUS_CNT has already been described. This signal is now labeled EN_DLL in the ADPLL timing diagram shown in figure 23a. Correction of the clock phase will take place once every 250μs during the time EN_DLL is active. The 15.36MHz reference clock is used to clock the ADPLL. The objective of the ADPLL is to push or pull the edge of the clock signal, MCLK_RX, in time until it coincides with the middle of each S bus symbol. The resolution of the ADPLL is limited to 65ns +/- 100ppm. This is the period and resolution of the 15.36MHz reference crystal. A positive edge detector samples the signal, EN_DLL to produce EN_DLL_EDGE. This starts the ADPLL clock correction process once synchronisation with the S bus is established. The signal, FL_PAIR, from the FIX_POLARITY_ARCH represents the L symbol following the F symbol. An edge detector clocked by the 15.36MHz clock samples FL_PAIR. When the edge of FL_PAIR is identified the signal, EN_DLL_EDGE, is activated. If the clock signal, MCLK_RX is inverted to create MCLK_FB and fed through a positive edge detector then the signal MCLK_FB_EDGE identifies the edge of MCLK_RX.

A state machine compares the signals FL_PAIR_EDGE and MCLK_FB_EDGE and when the two signals are aligned the system is locked. This condition is shown in figure 23a, and it can be seen that the sampling clock is now coincident with the center of the S bus symbols. When MCLK_FB_EDGE lags FL_PAIR_EDGE the signal, MCLK_RX, is pushed in time to catch up with the

edge of the FL pair. This is shown in figure 23b. When MCLK_FB_EDGE leads FL_PAIR_EDGE the signal, MCLK_RX, is pulled in time as shown in figure 23c.

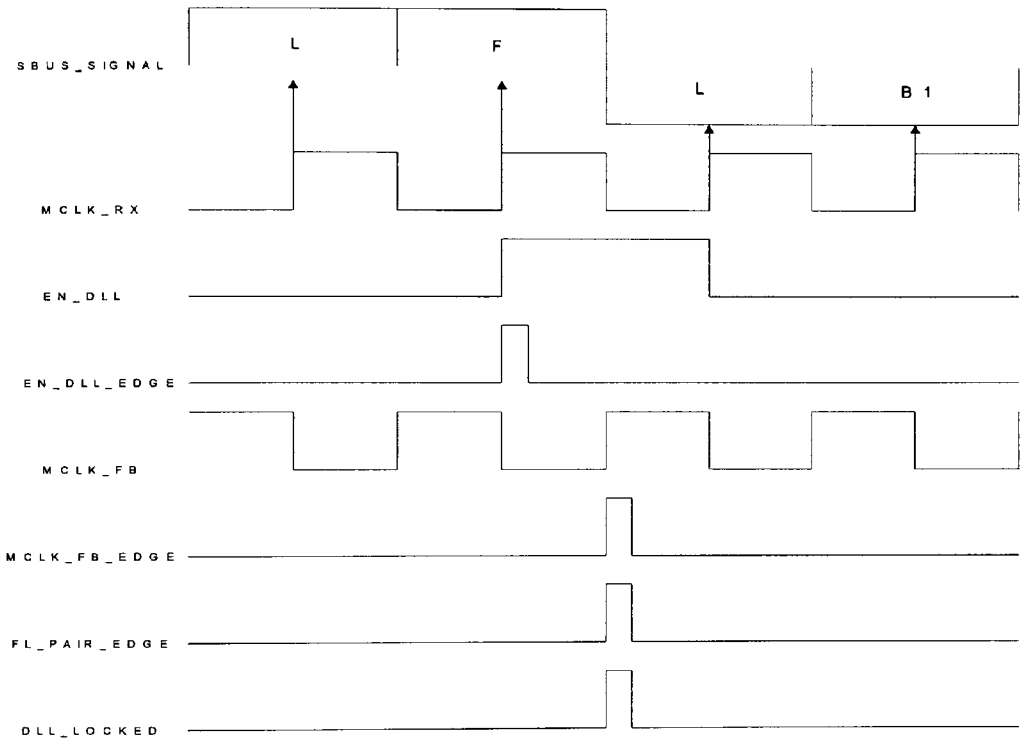


Figure 23a - The ADPLL timing diagram.

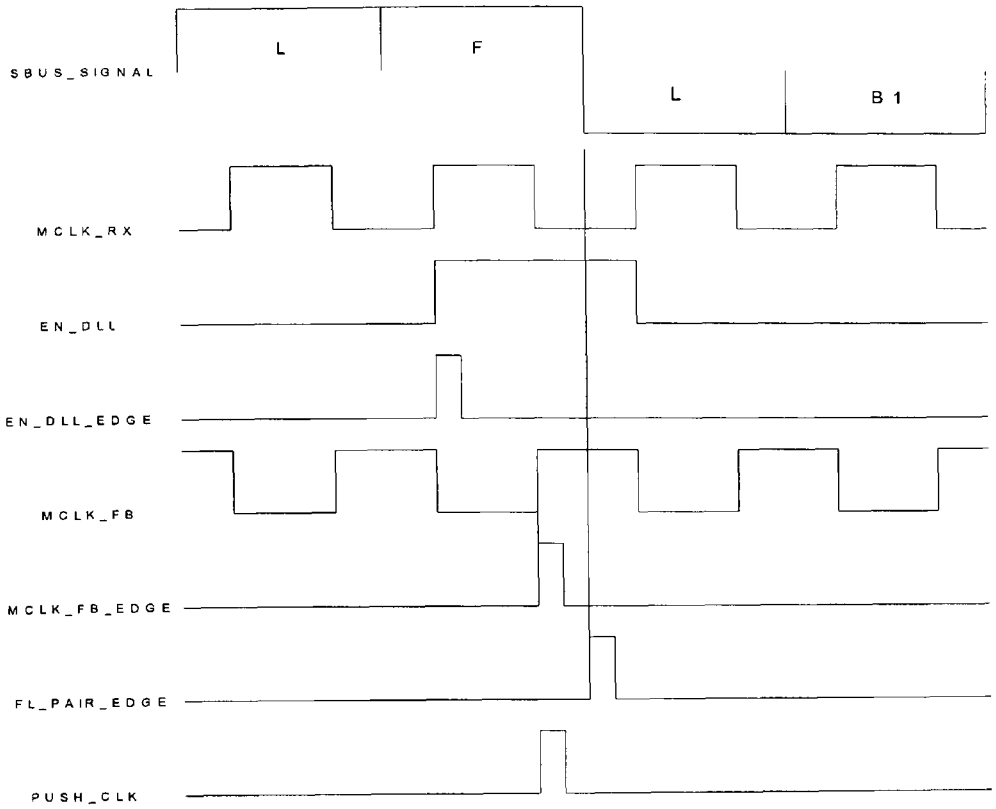


Figure 23b - The ADPLL timing diagram.

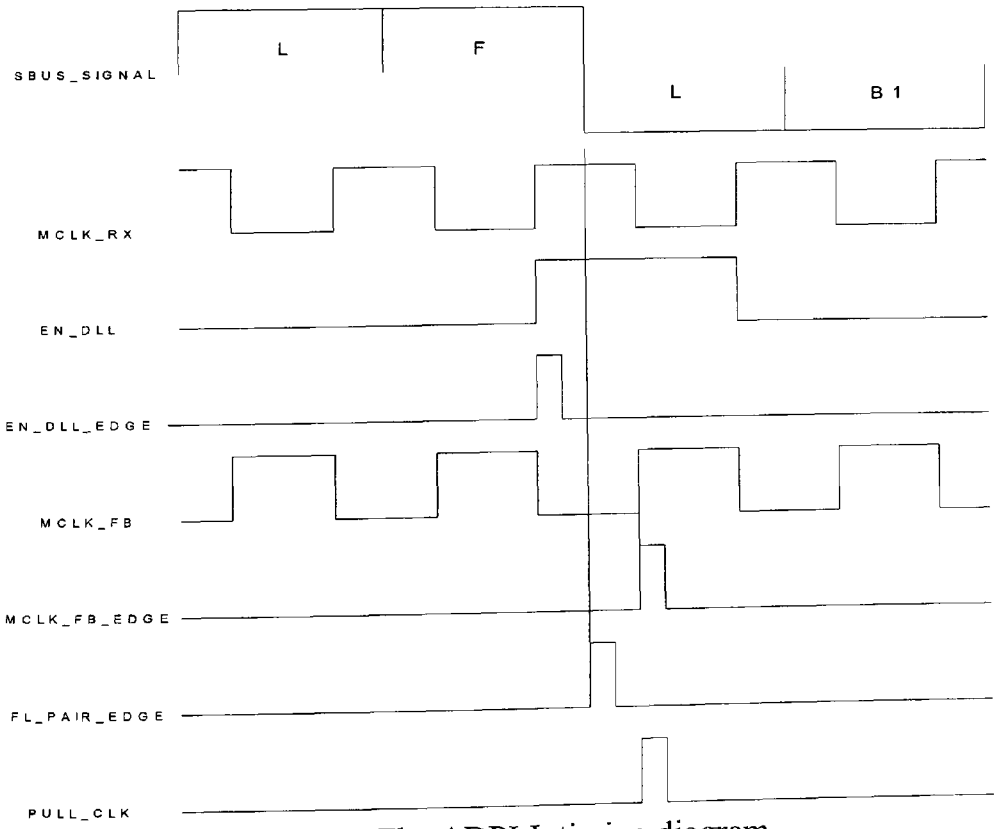


Figure 23c - The ADPLL timing diagram.

The ADPLL block diagram is shown in figure 2 of appendix a. Seven sub-blocks can be identified. The block diagram is composed of the DLL_CNT, DLL_OFFSET_CNT, DLL_COMP, DLL_ARCH, DLL_CON, and KICKSTART functional blocks. The following describes the operation of each block in turn. The clocks for the Terminal equipment are derived from the ADPLL block. The clock MCLK_RX is used to sample data from the S bus. The clock MCLK_TX is the inverse of MCLK_RX and is used to transmit data onto the S bus. These two clocks are buffered with dedicated clock buffers (BUFG). This minimizes clock skew when the clock is routed across the FPGA. When the FPGA is reset the KICKSTART block monitors the next pulse on the S bus and forces MCLK_RX to coincide with the center point of each symbol on the S bus. This will be discussed in more detail in the following section.

The MCLK_GEN block.

The MCLK_GEN block implements a toggle flip-flop. When the block is reset the output signal, MCLK goes low. When the input signal TOGGLE_MCLK is asserted high for one period of the clock signal, 15_36MHZ then MCLK changes state. The signals, MCLK_TX and MCLK_RX are derived from MCLK. The MCLK frequency is 192kHz or 5.2μs with 50% duty cycle. 5.2μs divided by 65ns is 80. A counter with 40 states is clocked with the 15.36MHz clock to produce the signal, TOGGLE_MCLK. TOGGLE_MCLK goes active every 2.6μs. The timing diagram for the operation is shown in figure 24.

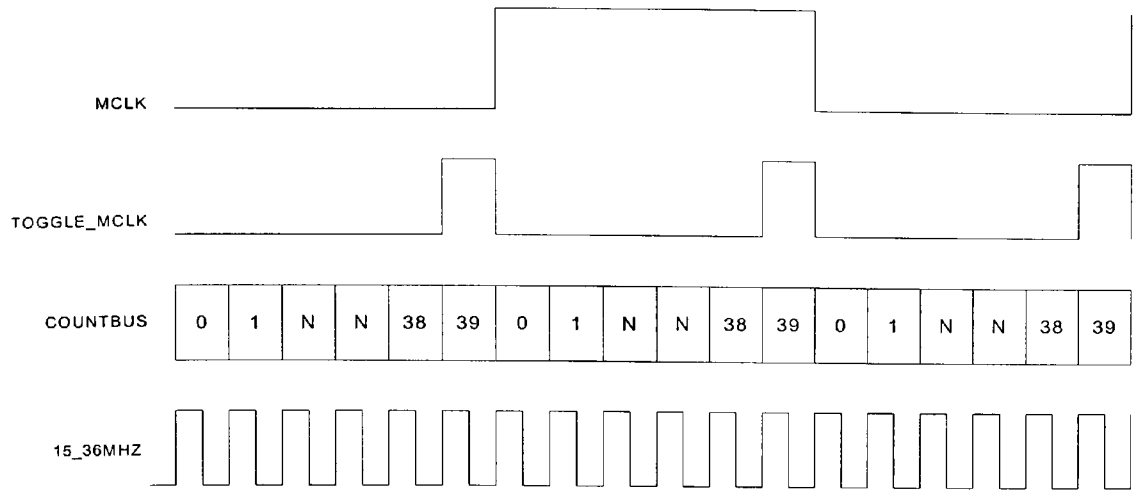


Figure 24 - The timing diagram for the MCLK clock signal.

The DLL_CNT block.

The DLL_CNT block generates the count sequence represented by COUNTBUS in figure 24. The block is clocked with the 15.36MHz reference clock signal. COUNTBUS is a 6-bit bus representing the binary count. The signal RST synchronously resets the count sequence to 0. The signal, KICKSTART also resets the count sequence to 0.

The DLL_OFFSET_CNT block.

The DLL_OFFSET_CNT block implements an up/down counter. The counter is clocked with the 15.36MHz reference oscillator. The counter has 40 states, 0 to 39, and when reset the counter assumes state number 39. The counter advances the state sequence when the signal PUSH is active. If the counter state is 39, an active PUSH signal will advance the count sequence to 0. The counter counts down from the current state when the signal PULL is active. If the counter state is 39, an active PULL signal will decrement the count sequence to 38.

The DLL_COMP block.

The DLL_COMP block compares the count sequence from the DLL_CNT block and the DLL_OFFSET_CNT block. The output signal TOGGLE_MCLK is asserted high for one period of the 15.36MHz clock when both count sequences are the same.

The KICKSTART block.

In order to reliably sample data from the S bus it is desirable to sample at the center point of each symbol. The KICKSTART block works with the DLL_CNT block, the DLL_OFFSET_CNT block, the DLL_COMP block and the MCLK_GEN block to achieve this. The timing diagram in figure 25 shows the arrangement. When the reset signal clears the FPGA the KICKSTART block is armed. In this state the KICKSTART block monitors all activity on the S bus and issues a reset to the DLL_CNT block, the DLL_OFFSET_CNT block and the DLL_COMP block when a pulse is received from the NT. This sets the OFFSET count sequence to 39. The COUNTBUS sequence is reset to 0 and the count sequence advances with each 15.36MHz clock cycle until state 39 is reached. At this point both count sequences are the same and the signal TOGGLE_MCLK sets MCLK high. MCLK is now coincident with the middle of the pulse on the line. The toggle action will continue setting and resetting MCLK at this point until synchronisation is achieved. The ADPLL is then enabled and if the edge of MCLK is not aligned with the FL pair edge an adjustment is made.

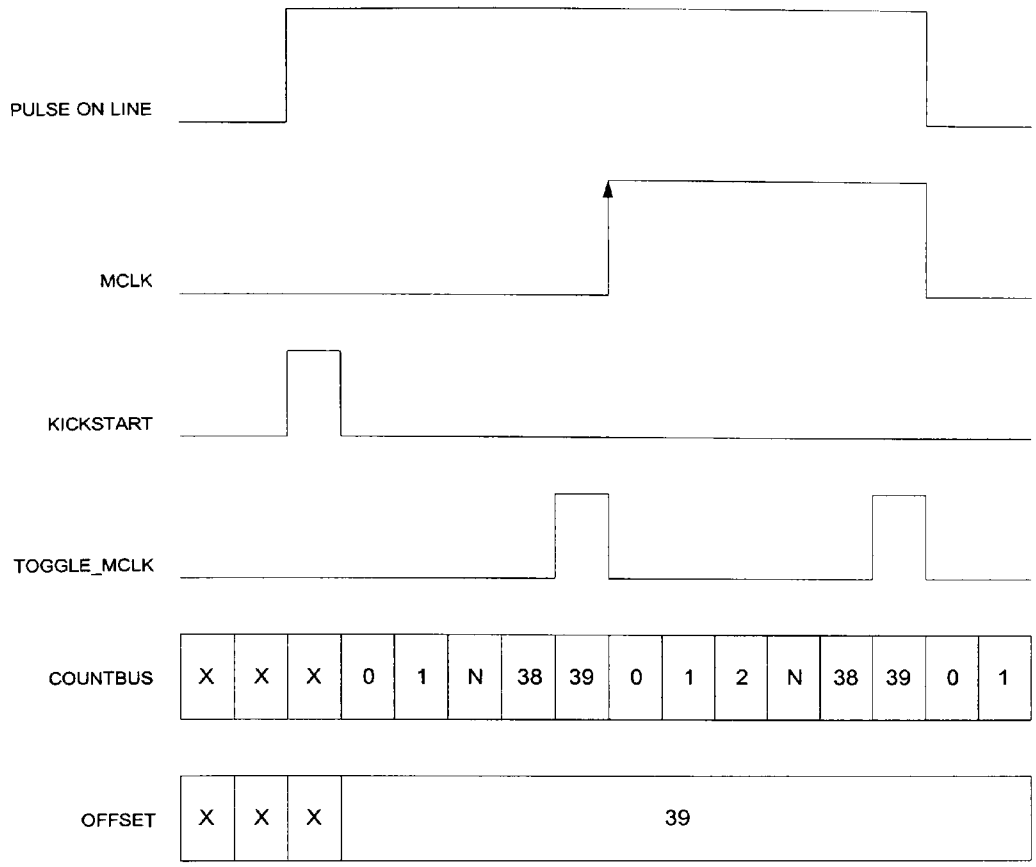


Figure 25 - The KICKSTART timing diagram.

The DLL_ARCH block.

The DLL_ARCH block implements a positive edge detector circuit. Each of the input signals EN_DLL, FL_PAIR, and MCLK_FB are inputs to a separate edge detector. The outputs ENDLL_EDGE, FLPAIR_EDGE and MCLKFB_EDGE occur synchronously with the 15.36MHz clock signal.

The DLL_CON block.

When synchronisation is achieved with the S bus the DLL_CON block is enabled. The DLL_CON block implements the state machine for the ADPLL. The timing diagrams already presented in figure 23a, 23b and 23c explain how the ADPLL

operates. The signals PUSH_CLK and PULL_CLK advance or decrement the counter DLL_OFFSET_CNT. This in turn pushes or pulls the edge of MCLK relative to the S bus FL pair timing event. To demonstrate the process, assume that the last state of DLL_OFFSET_CNT was 39 and the signal PUSH_CLK has advanced the counter to the current state 0. Figure 26 presents the result. The signal TOGGLE_MCLK has now moved forward in time from the old state (shown in dashed lines). The MCLK will now change state 65ns later as shown. The clock pulling operation operates in a similar fashion. The state machine ASM chart is shown in figure 27. The states are describes as follows.

STATE S0: This is a wait state and when the ENDLL_EDGE goes active the state sequence moves the S1.

STATE S1: The decision to adjust the clock is made in S1. In state S1 the edge of MCLK (MCLKFB_EDGE) is compared to the edge of the FL pair (FLPAIR_EDGE).

STATE S2: The signal PUSH_CLK goes active for one state.

STATE S3: The signal PULL_CLK goes active for one state.

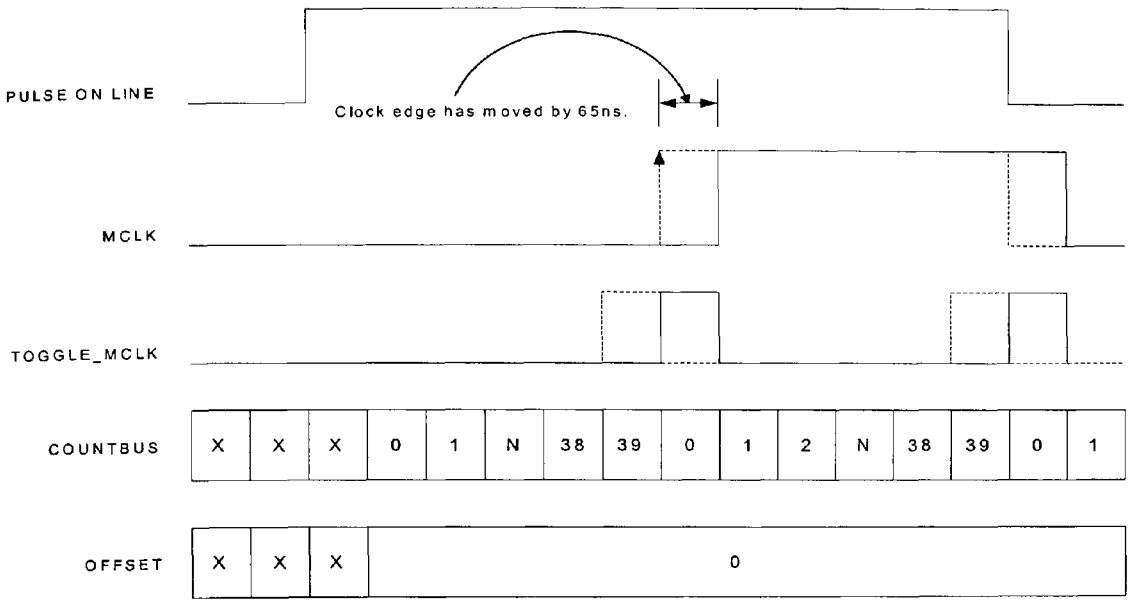


Figure 26 - The timing diagram for the ADPLL.

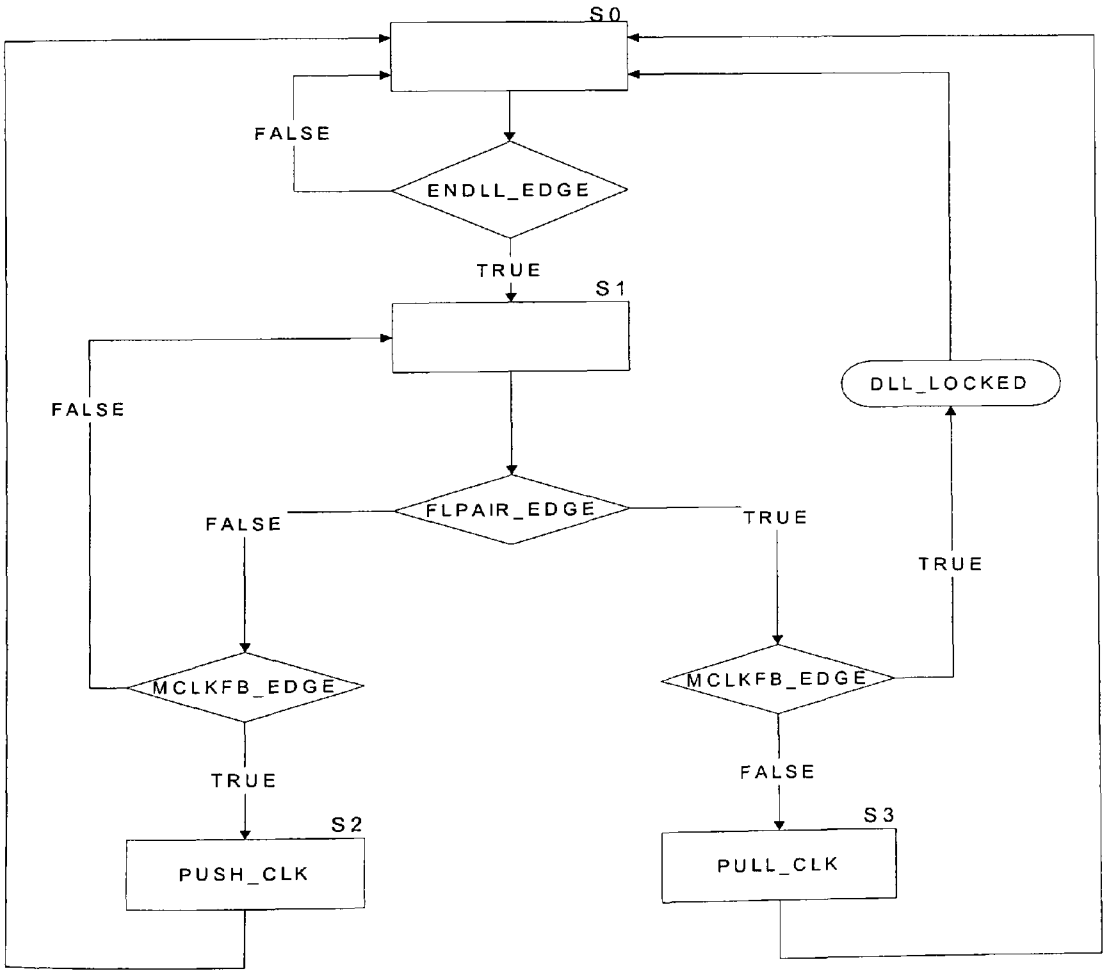


Figure 27 - The ASM chart for the DLL_CON block.

4.3 The design validation of the SBUS_DLL block and measured results.

The design validation stage focused on the verification of the synchronisation process and the clock phase locking process. The jitter of the extracted clock is measured and compared to the requirements stated in recommendation I.430. In the following section, oscilloscope plots of the circuit operation captured during the design validation are presented. The clock frequency of both MCLK_RX and MCLK_TX is shown in figure 28. The clock frequency is measured as 5.2μs. The MCLK_RX and MCLK_TX clocks are used to clock the circuits that extract data from the S bus and transmit data onto the S bus. Therefore jitter on the clocks will directly affect the jitter in the TE output signal. It is desirable to keep the clock jitter below that required by the system The timing extraction jitter requirement, as stated in I.430, is +/- 7% of a bit period. The bit period in this case is 5.2 μs and thus the clock jitter is 364ns. The measured clock jitter as shown in figure 29 is 160ns or 3% of a bit period.

The plot in figure 30 shows the signal from the NT on channel 2 and the sampling clock MCLK_RX on channel 1. The oscilloscope was set to trigger from channel 2 and MCLK_RX was then observed. It was found that MCLK_RX remained stationary with the positive edge coincident with the middle of each S bus symbol as expected. Figure 31 shows the signal from the NT on channel 1 and the signal EN_DLL, generated by the TE is shown on channel 2. The FL pair symbol on channel 1 can be identified as the negative pulse immediately followed by the positive pulse. The signal EN_DLL should consistently occur synchronously with the sampling clock after the first violation is identified. The edge of the signal EN_DLL should remain within 2.6μs from the edge of the L symbol (shown as the

first positive pulse on channel 1) while synchronisation is established and while the clocks are locked. This was confirmed to be the case. The EN_DLL signal enables the phase locked loop. It is necessary to compare more that two signals and as the oscilloscope has only two channels. Cross-referencing of signals will be made using the EN_DLL signal as a measurement reference. The timing diagram for the ADPLL was presented in figure 23. The objective of the following measurement is to determine the actual ADPLL timing. A comparison with figure 23 will be made and a conclusion drawn. The following measurements were taken when synchronisation was established and the ADPLL was working. Figure 32 shows the MCLKFB_EDGE signal on channel 1 and EN_DLL on channel 2. According to the timing diagram in figure 23 the MCLKFB_EDGE is located in the correct position. The signal FLPAIR_EDGE is expected to move relative to MCLKFB_EDGE. The vertical cursors in figure 33a were set to window the MCLKFB_EDGE signal in figure 32. FLPAIR_EDGE was then measured with channel 1. The objective was to compare the signals MCLKFB_EDGE and FLPAIR_EDGE. Figures 33b and 33c clearly show how the EN_DLL signal, derived from the system timing moves about the FL pair on the S bus. Observing this over time concluded that the ADPLL was adjusting the phase of the MCLK_RX in order to maintain alignment with FLPAIR_EDGE. Figure 34 shows the DLL_LOCKED signal active indicating that the ADPLL is working correctly.

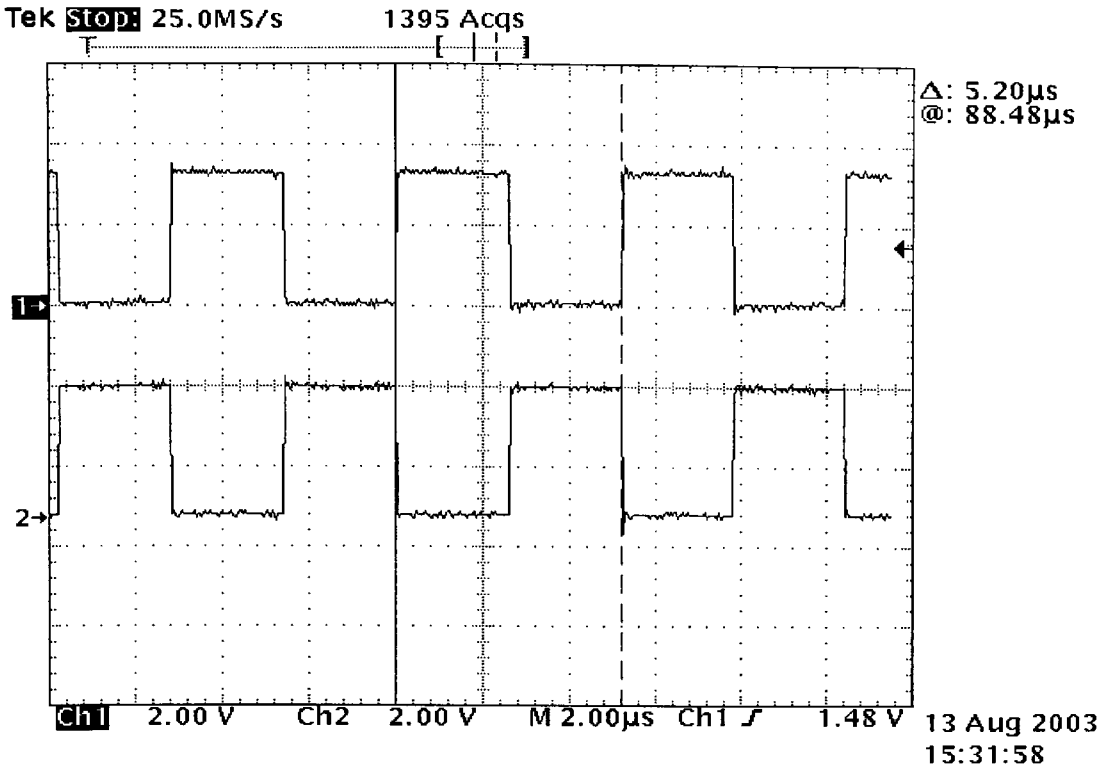


Figure 28 - The clock frequency of MCLK_RX and MCLK_TX.

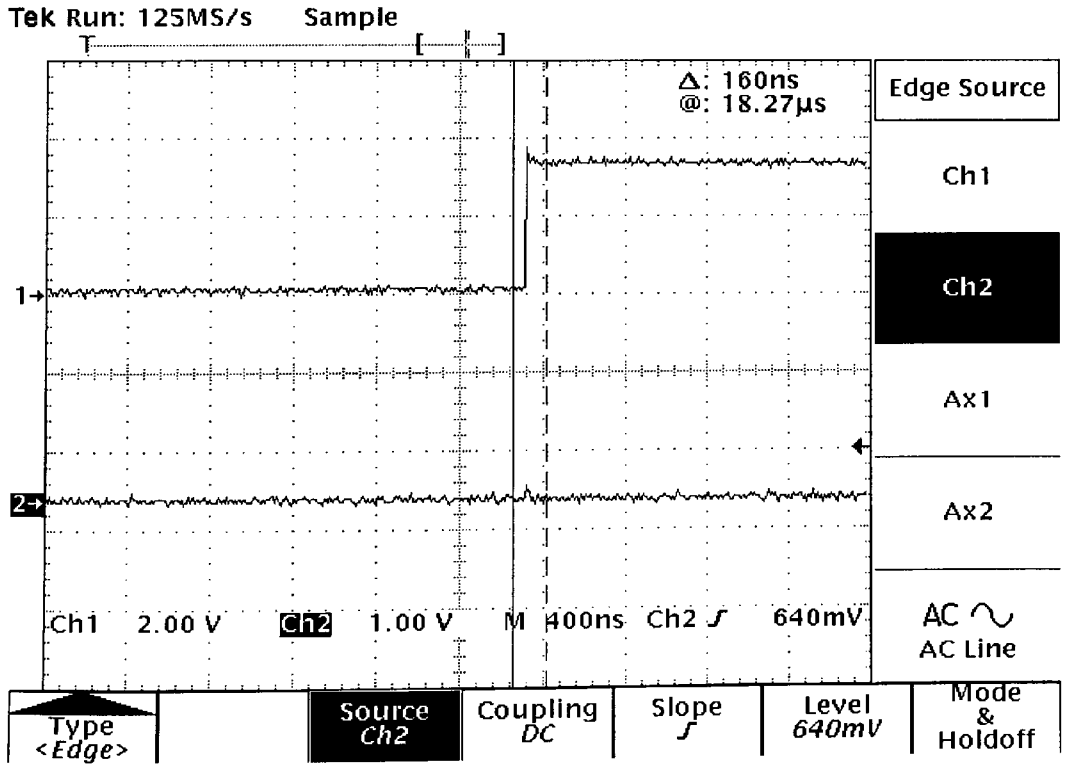


Figure 29 - The Jitter measurement.

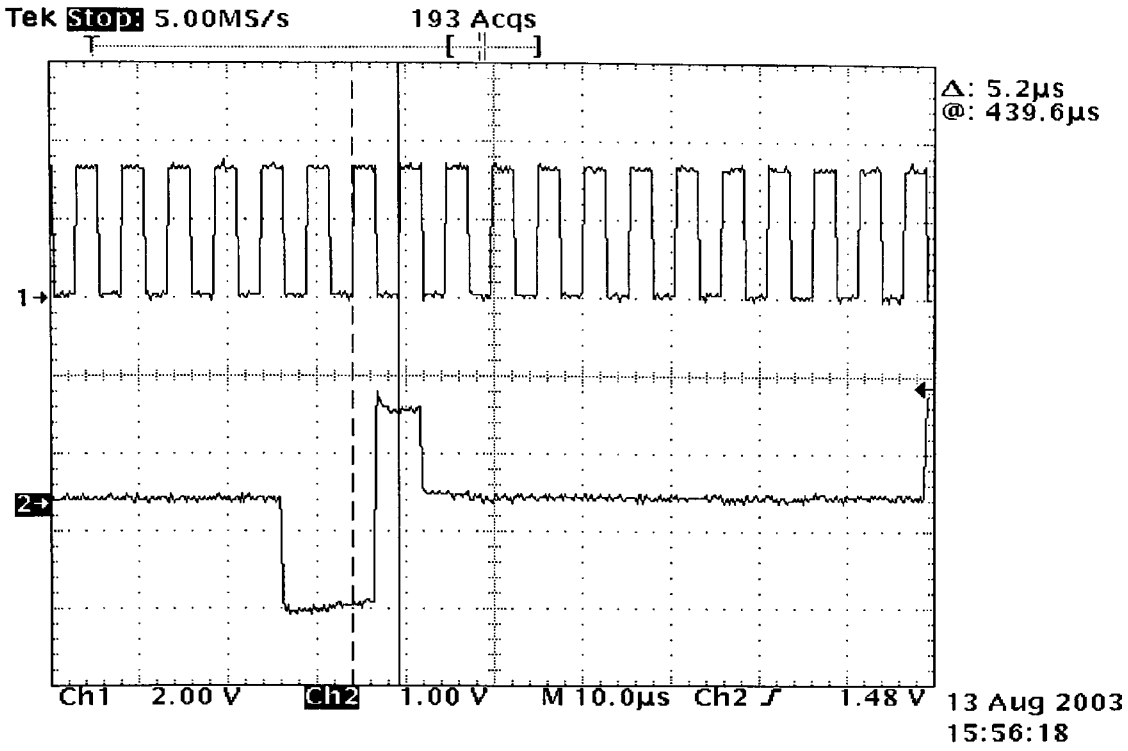


Figure 30 - The clock signal MCLK locked to the NT transmitted signal.

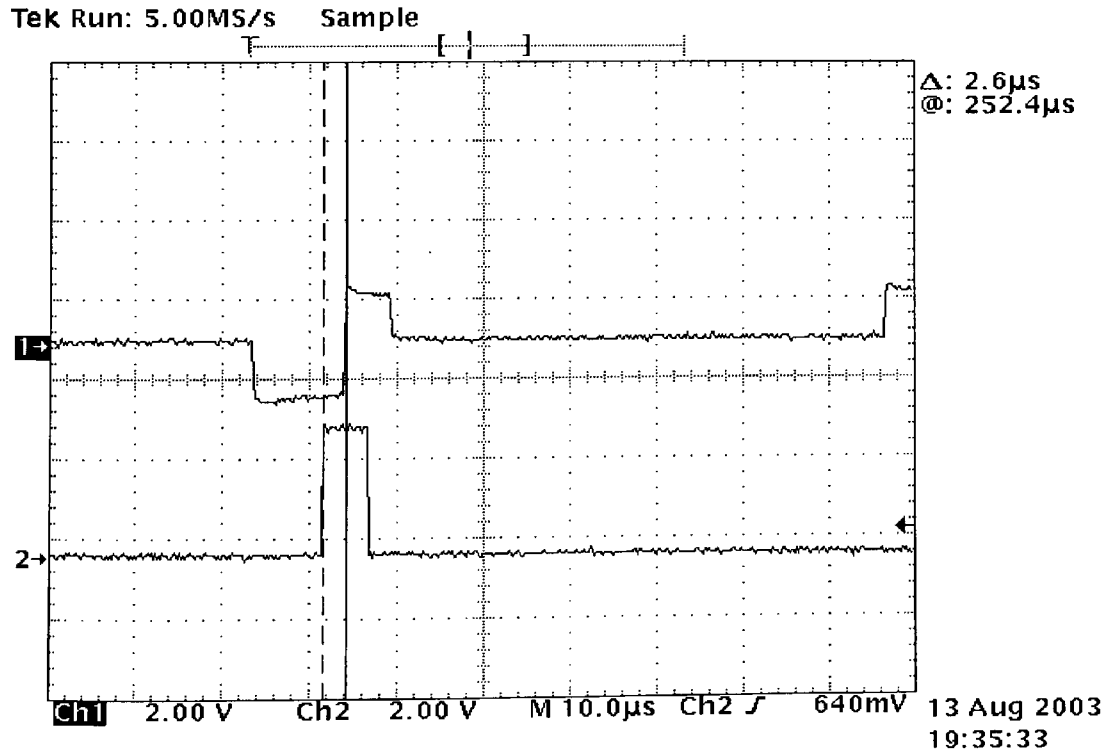


Figure 31 - The EN_DLL signal.

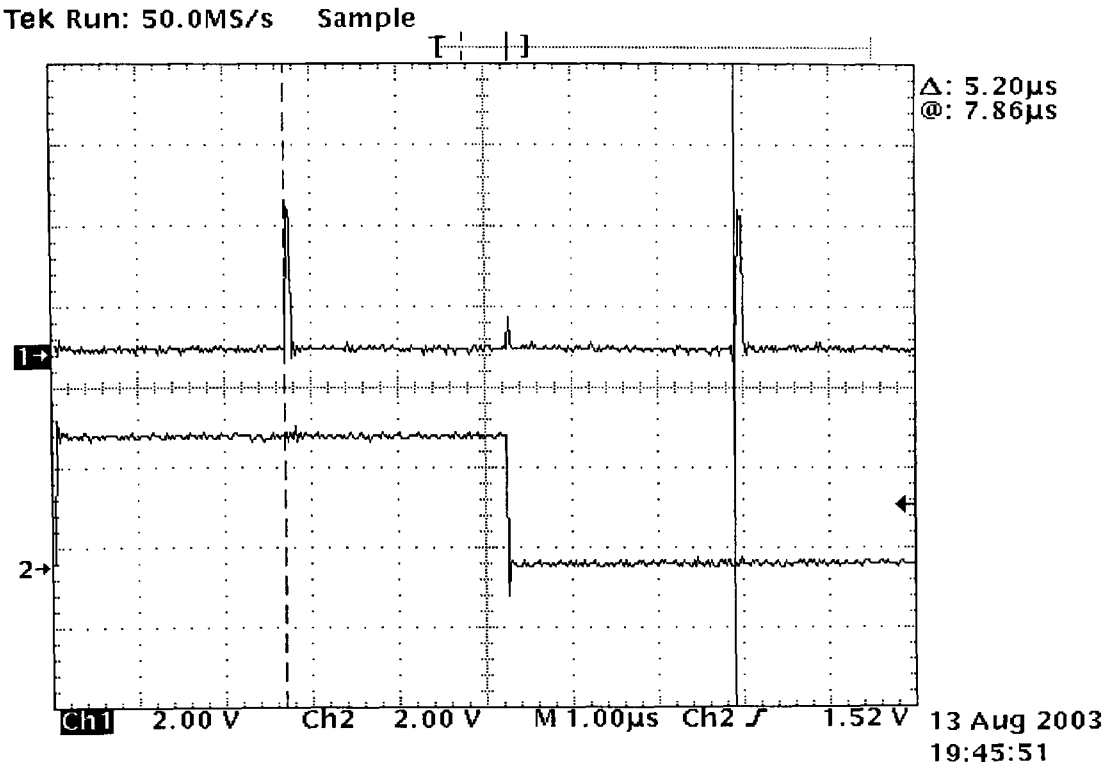


Figure 32 - The MCLKFB_EDGE signal.

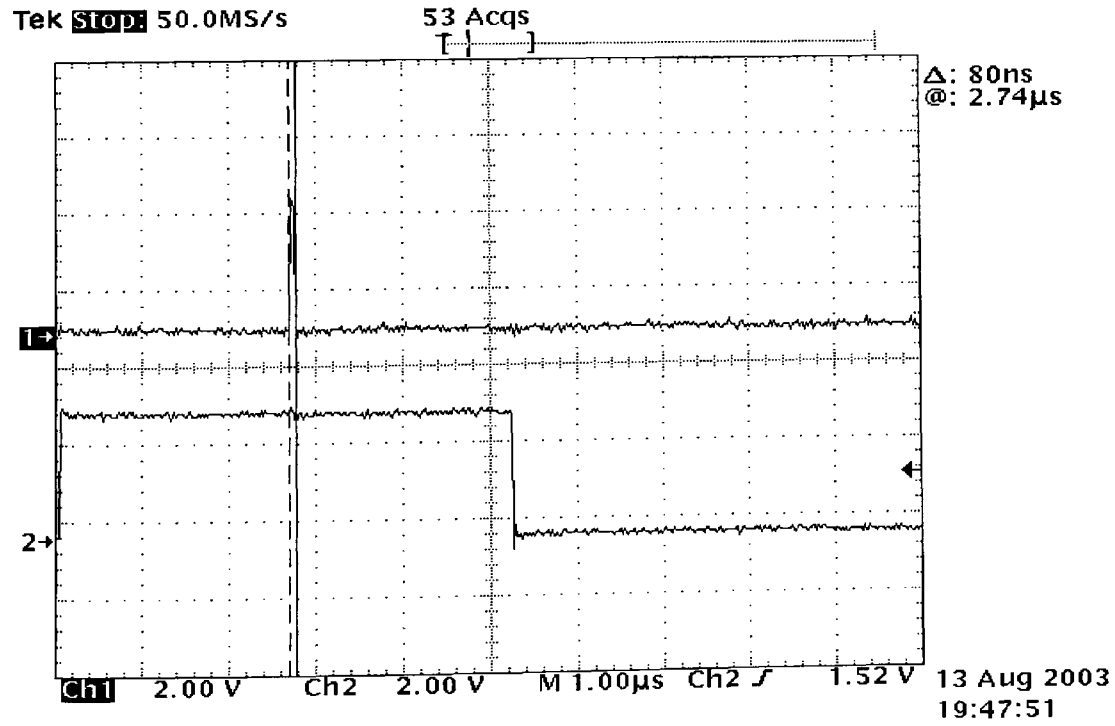


Figure 33a -The FLPAIR_EDGE signal.

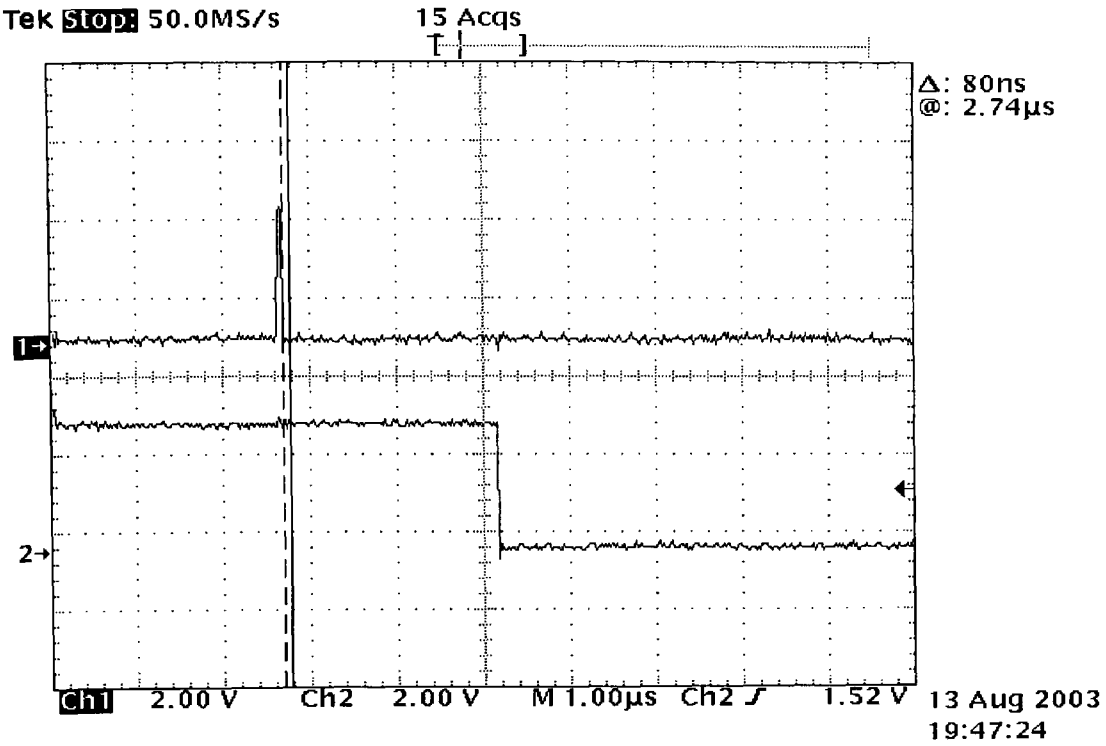


Figure 33b -The FLPAIR_EDGE signal.

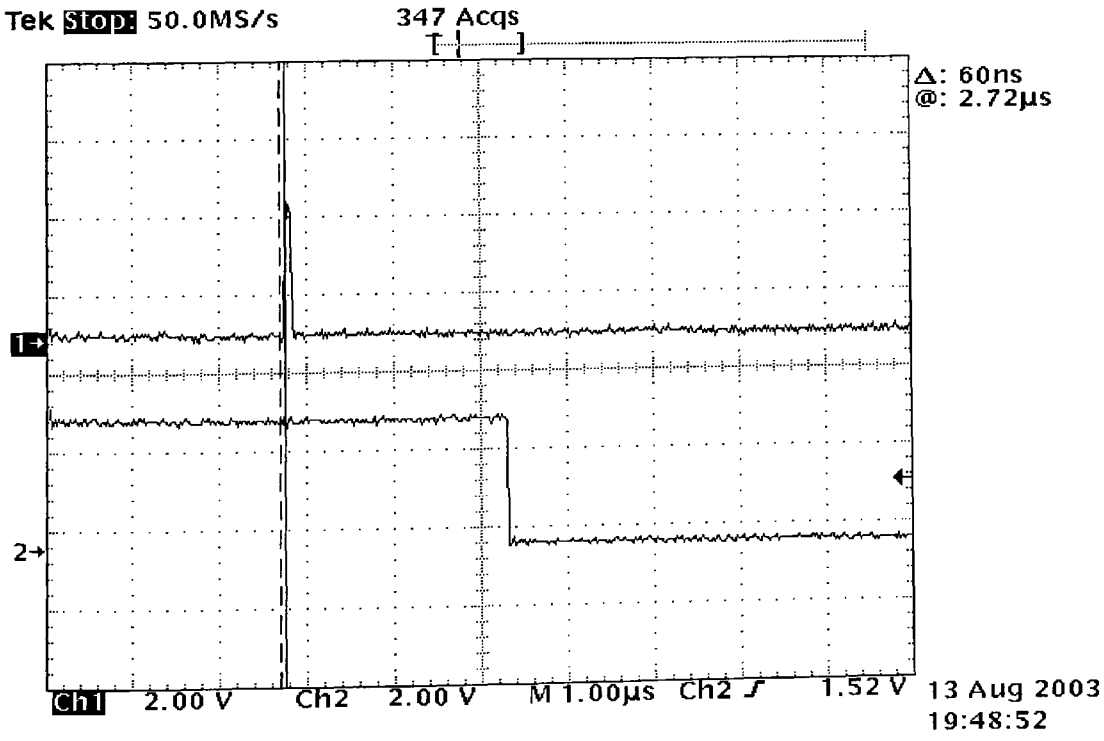


Figure 33c -The FLPAIR_EDGE signal.

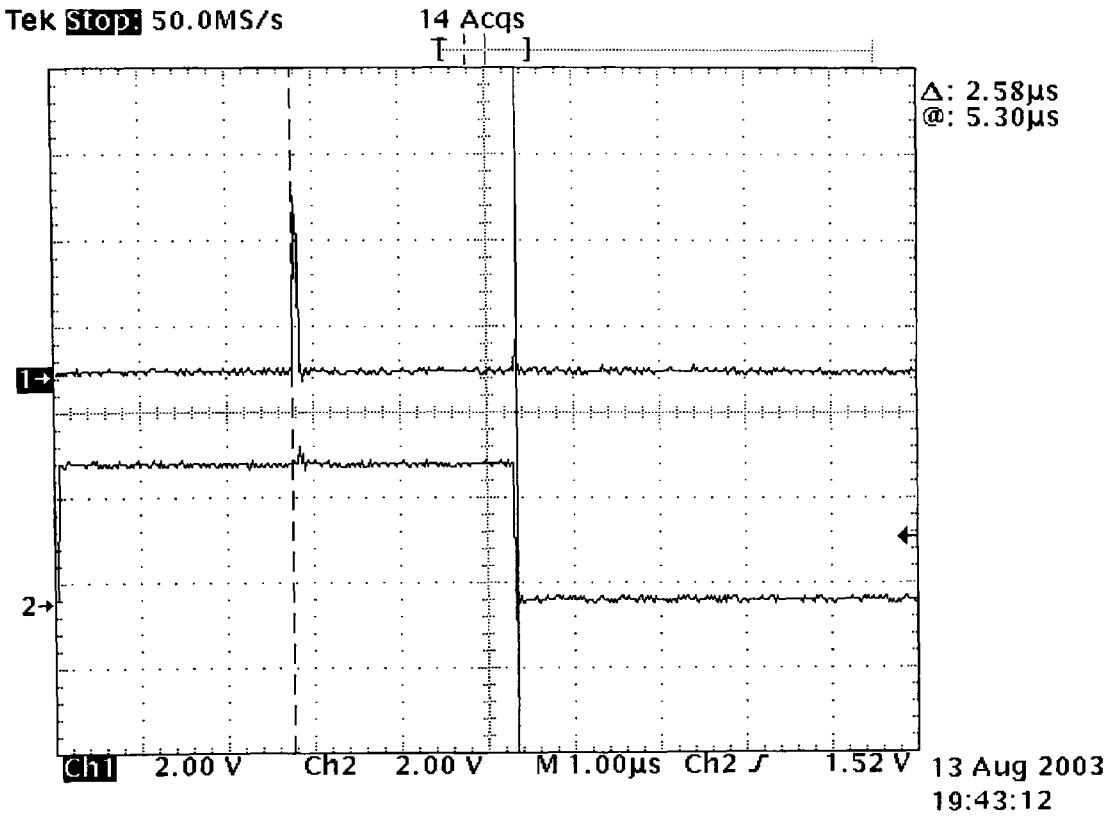


Figure 34 - The DLL_LOCKED signal.

4.4 The ISDN Signal generator for the TE to NT direction.

The ISDN Signal generator performs the signal coding and data framing of B channel data and Q channel data for onward data transmission to the NT according to the I.430 specification. The ISDN signal generator is implemented by the INFOGEN_TENT block and was introduced in chapter 4. The INFOGEN_TENT block diagram is presented in figure 3 of appendix a. The block diagram identifies two main sub-blocks in the design, INFO1_GEN and INFO3_GEN. The INFO1_GEN and the INFO3_GEN blocks generate the signal types required for data transmission in the TE to NT direction. The INFO1_GEN block generates an INFO1 signal. The INFOGEN_DECODE block produces the signals, LATCH_BBITS and LATCH_DBITS when the count sequence for the TE to NT direction of transmission reads 46. The INFO3_GEN generates the INFO3 signal. The INFOGEN_TENT block will send INFO1 until a synchronisation indication is received from the SBUS_DLL block. When synchronisation is established the INFO3 signal is sent. The INFOGEN_TENT block receives B channel and Q channel data and stores it internally for onward transmission in the next data frame to the NT. The block is clocked by MCLK_TX, which is generated by the SBUS_DLL block. Data transmission onto the S bus is synchronised to the count sequence, SBUSCNT_TENT, which is received from the SBUS_DLL block.

4.4.1 The INFO1_GEN block.

The INFO1_GEN block generates the INFO1 signal. The block diagram for the system is shown in figure 4 of appendix a, and identifies the following two sub-blocks. The timing diagram for system is shown in figure 35. The resulting INFO1 signal on the line is shown and is composed of two pulses of alternate polarity followed by six spaces. The control signals required by the differential amplifier driver circuit to produce this signal are POS_PULSE and NEG_PULSE. The order in which the pulses appear is defined by the state-machine implemented by the INFO1_CON block. The state sequence is shown in figure 36. The process involves generating the POS_PULSE signal, then the NEG_PULSE signal and then waiting for six clock cycles before repeating the process. The system is clocked by the signal CLK (MCLK_TX from the SBUS_DLL block). The signal ADV_BITCNT enables a binary count sequence in the INFO1_ARCH sub-block. The INFO1_ARCH sub-block implements the binary count sequence BIT COUNT shown in the timing diagram. The count sequence is decoded and the signal BITCNT_UP is produced on the sixth state. The state machine then resets the count sequence with the signal RST_BITCNT and the process repeats. The actual INFO1 signal generated by the system can be seen in figure 14.

The INFO1_CON.

The ASM chart for the INFO1_CON is shown in figure 36.

The INFO1_ARCH block.

The functionality of the INFO1_ARCH block has already been described.

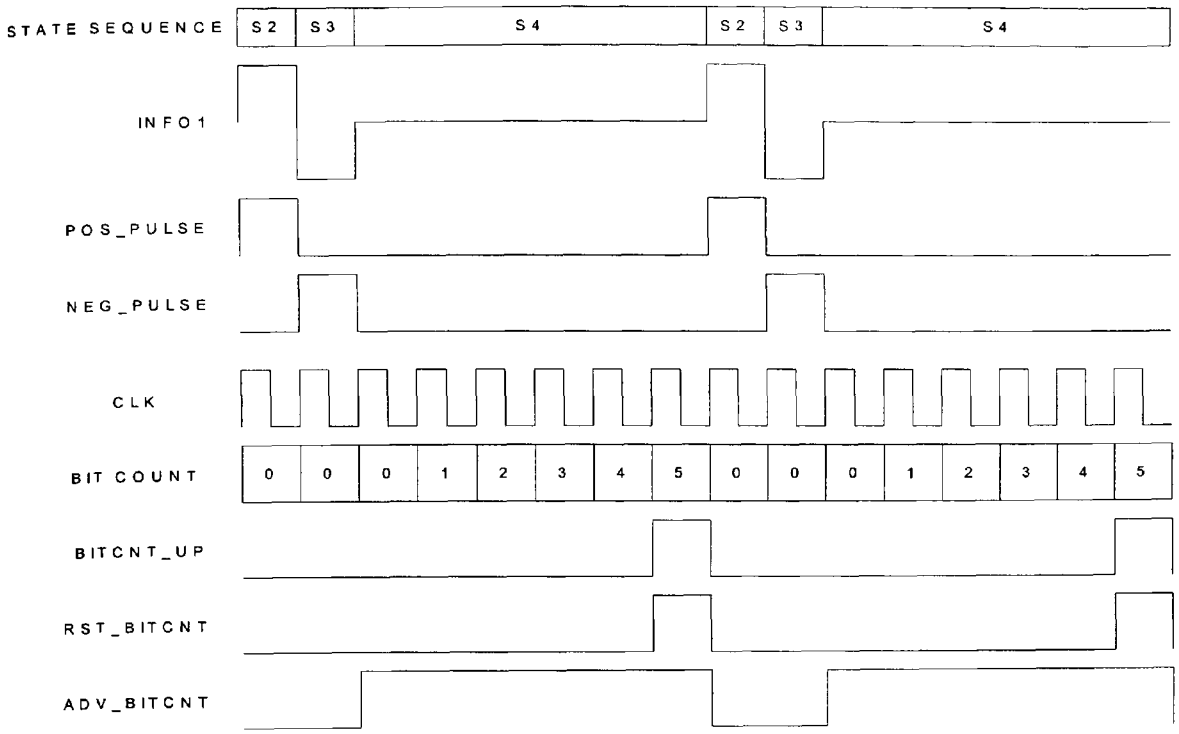


Figure 35 - The timing diagram for the INFO1_GEN block.

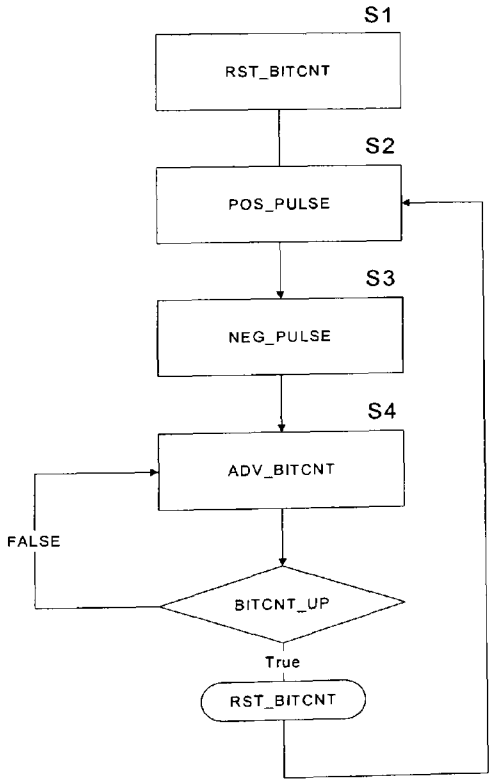


Figure 36 - ASM chart for the INFO1_CON block.

4.4.2 The INFO3_GEN block.

The INFO3_GEN block performs the data framing and AMI signal encoding in the TE to NT direction of transmission. B channel, D channel and the Q channel encoding are supported by the design. While the D channel is fully supported, the implementation ensures that logic ones are sent in the D channel to disable it. The INFO3_GEN block diagram can be seen in figure 5 of appendix a, and is entitled INFO3_GEN. The system is composed of a number of sub-blocks. Before these are described a general overview of the system will be presented.

The INFO3_GEN block generates the coding violations according to the coding rules outlined in I.430. The block is clocked with the MCLK_TX clock from the SBUS_DLL block. There should be an offset of two bits between the data bits received from the NT and the corresponding data bits in the TE to NT direction. The count sequence, SBUS_COUNT_TENT, generated by the SBUS_DLL block identifies each symbol on the S bus toward the NT. The relationship between the count sequence and the associated symbols on the SBUS forms an important design convention. All the blocks in the Terminal equipment design follow this convention. The SBUS_COUNT_TENT count sequence has already been defined and following the convention will ensure the 2-bit offset between NT and TE data frames is maintained. Once synchronisation is established with the NT the INFO3 signal generation begins. The timing diagram presented in figure 37 shows the relationship between data frames in both directions of transmission.

The discussion from here on will identify bits in the TE to NT direction by the state of the count sequence SBUS_COUNT_TENT. The F bit in this case would be referred to as bit 47. Bit 45 of the TE to NT frame is decoded to produce the

signal START_NEXT_FRAME. The signal START_NEXT_FRAME will go active every 250μS and will be used to control the signal coding process.

The INFO3_GEN block processes data on a frame-by-frame basis. The INFO3_GEN block loads sixteen bits of B1 data and sixteen bits of B2 data into the system once per frame. This action takes place when the SBUS_COUNT_TENT count sequence reads 46. In the same way, the system loads four bits of D channel data once per frame. It is the responsibility of the electronics downstream to ensure that new data is available on the B channel input data-bus and the D channel input data-bus before the SBUS_COUNT_TENT count sequence reads 46. If there is no data to be transmitted logic 1s must be placed on the input data busses. In this case the D channel is not used and the D channel input bus is wired permanently to logic 1.

The maintenance data is sent to the NT in the Q channel. The Q channel is embedded within the data frame. Maintenance commands on the Q channel are organised as 4-bit words. The Fa bit in the TE to NT direction doubles as a Q bit once every four frames. Therefore, it takes four frames to send one maintenance command to the NT. There is a relationship between the maintenance (S channel) channel in the NT to TE direction of transmission and the Q channel in the TE to NT direction of transmission. Synchronisation with the S channel must first be established before the location in time of the Q channel can be determined. It is the responsibility of the downstream electronics to perform this synchronisation task and indicate to the INFO3_GEN block when it is time to send a Q bit. Once the downstream electronics has synchronized to the S channel from the NT, the structure of the Q channel will be known. It is the responsibility of the downstream electronics to refresh the data to be sent on the Q channel.

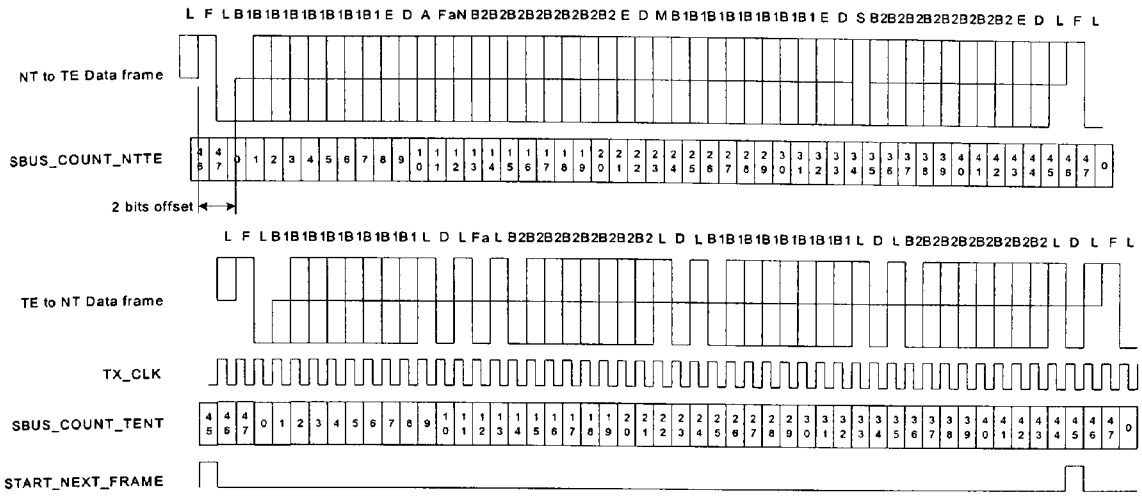


Figure 37 - The relationship between SBUS data frames.

The INFO3_GEN input signals.

CLK: This is the clock signal for the system. It is connected to the MCLK_TX clock.

RST: This is a synchronous reset signal for the design.

SYNC: This signal is from the SBUS_DLL block and indicates that synchronization to the NT has been established and INFO3 generation can begin.

EN_INFO3: This signal enables INFO3 generation when set to logic 1.

LATCH_DBITS: This latches four bits of D channel data into system.

LATCH_QBITS: This latches four bits of Q channel data into system.

LATCH_B2BITS: This latches sixteen bits of B2 channel data into system.

LATCH_B1BITS: This latches sixteen bits of B1 channel data into system.

Q[3:0]: This is the four bit Q channel input data bus.

D[3:0]: This is the four bit D channel input data bus.

B2[15:0]: This is the sixteen bit B channel input data bus.

B1[15:0]: This is the sixteen bit B channel input data bus.

QBIT: This indicates that the Fa bit in the current frame is a Q bit.

SBUSCNT_TENT: This is a count sequence for the data frame in the TE to NT direction of transmission. The count sequence identifies each bit within the frame.

The INFO3_GEN output signals.

POS_PULSE: This signal is a control signal to the differential driver circuit in the analogue signal transmitter.

NEG_PULSE: This signal is a control signal to the differential driver circuit in the analogue signal transmitter.

The INFO3_GEN circuit architecture.

The circuit architecture provides the data storage for the information to be transmitted and facilitates the state machine in controlling the process. The sub-blocks for the INFO3_GEN are the INFO3_B1_REG, INFO3_4BIT_REG, SBUSCNT_DECODE_FOR_INFO3,INFO3_BBITS_MUX,INFO3_BSHFTSIGNAL_MUX, and the INFO3_ARCH. The following explanation details the circuit operation of each block.

The INFO3_B1_REG block.

The INFO3_B1_REG implements a sixteen bit parallel in/serial out shift register. The block is used twice. U5 provides storage for the B1 channel data and U6 services the B2 channel data. The following describes the inputs and outputs for the block.

The INFO3_B1_REG inputs.

CLK: This is the clock signal and is connected at the top level of the design hierarchy to MCLK_TX.

RST: This is a synchronous reset.

B1[15:0]: This is the sixteen bit input bus containing B channel data.

LATCH_DATA: This loads the data on the sixteen bit input bus into the shift-register.

SHFT_BBITS: This shifts the data in the shift register out serially on the NEXT_B output signal.

The INFO3_B1_REG outputs.

NEXT_B: Data is valid on this signal after the positive going edge of CLK.

The INFO3_4BIT_REG block.

The INFO3_4BIT_REG implements a four bit parallel in/serial out shift register. The block is used twice in the design. U2 is used to store D channel data and U3 is used to store Q channel data. The following describes the inputs and outputs for the block.

The INFO3_4BIT_REG inputs.

CLK: This is the clock signal and is connected at the top level of the design hierarchy to MCLK_TX.

RST: This is a synchronous reset.

DIN[3:0]: This is the four bit input bus containing either D or Q channel data.

LATCH DATA: This loads the data on the four bit input bus into the shift-register.

SHFT_BITS: This shifts the data in the shift register out serially on the NEXT_BIT output signal.

The INFO3_4BIT_REG outputs.

NEXT_BIT: This is the output signal.

The INFO3_BBITS_MUX block.

This block implements a multiplexer. The B channel serial data stream from U5 or U6 is routed through to the multiplexer output based on the state of the input signal SEL_BBITS_MUX.

INFO3_BBITS_MUX input signals.

NEXT_B1: This is the serial data stream consisting of B1 data.

NEXT_B2: This is the serial data stream consisting of B2 data.

SEL_BBITS_MUX: logic 0 on this signal selects B1 data and a logic 1 selects B2 data.

INFO3_BBITS_MUX output signals.

NEXT_BBIT: This is the output signal.

The INFO3_BSHFTSIGNAL_MUX block.

This implements a multiplexer that is used to route the shift signal from the state-machine to either the B1 or B2 shift-register. This allows the state-machine to use one state to process B1 and B2 data alternatively and thus reduce logic.

The INFO3_BBITS_MUX input signals.

SHFT_BBIT_REG: This signal is from the state machine.

SEL_BBITS_MUX: This is the multiplexer select signal. When set to low SHFT_BBIT_REG is routed through to the SHFT_B1BITS output. When set to high SHFT_BBIT_REG is routed through to the SHFT_B2BITS output.

The INFO3_BBITS_MUX output signals.

SHFT_B1BITS: This shifts the B1 data into the state machine for processing.

SHFT_B2BITS: This shifts the B2 data into the state machine for processing.

The SBUSCNT_DECODE_FOR_INFO3 Block.

This decodes the count sequence SBUSCNT_TENT and produces the following output signals.

START_NEXT_FRAME: This signal is active when the count sequence reads 45.

NEXT_BIT_L: This signal is active when the count sequence reads 8,21,32 and 43.

PROCESS_FA_BIT: This signal is active when the count sequence reads 11.

LATCH_BBITS: This signal is active when the count sequence reads 0.

The INFO3_ARCH Block.

The INFO3_ARCH block implements the status flags used by the state machine during the process. The flags provide the following functions. The violation made flag allows the state machine to record that two valid violations have been achieved. The last pulse negative flag allows the state machine to record that the last pulse sent was a negative pulse. The last pulse positive flag allows the state machine to record that the last pulse sent was a positive pulse. The select B channel multiplexer

flag allows the state machine to set the signal SEL_BBITS_MUX, which in turn configures the architecture to route either B1 or B2 data to the state machine for processing. The Q bit counter allows the state machine to keep track of the number of Q bits sent. The inputs to the system are described as follows.

The INFO3_ARCH block inputs.

SET_VIOLATION_MADE: This sets the violation made flag to logic 1.

CLR_VIOLATION_MADE: This clears the violation made flag to logic 0.

SET_LASTPULSE_NEG: This sets the last pulse negative flag to logic 1.

CLR_LASTPULSE_NEG: This clears the last pulse negative flag to logic 0.

TOGGLE_BBITS_MUX: This changes the state of the B channel multiplexer flag.

CLR_BBITS_MUX: This set the B channel multiplexer flag to its default state, logic 0.

ADV_QBITS_CNT: This advances the state of the Q bits counter to the next state.

CLR_QBITS_CNT: This resets the Q bit counter to 0.

SET_LASTPULSE_POS: This sets the last pulse positive flag to logic 1.

CLR_LASTPULSE_POS: This clears the last pulse positive flag to logic 0.

The INFO3_ARCH block outputs.

VIOLATION_MADE: This is the output of the violation made flag. Two valid violations have been successfully made when this signal is set to logic 1.

LASTPULSE_NEG: This is the output of the last pulse negative flag. This indicates that the last pulse transmitted was a negative pulse when set to logic 1.

LASTPULSE_POS: This is the output of the last pulse positive flag. This indicates that the last pulse transmitted was a positive pulse when set to logic 1.

SEL_BBITS_MUX: This is the output of the B channel multiplexer flag. This selects the B1 channel data stream when set to logic 0.

The INFO3_CON block.

INFO3_CON implements a state machine to control the line coding process. The state machine has eleven states and each state handles the coding of specific bits within the data frame. Each state is numbered 1 through 11 and the timing diagram in figure 38 shows the active state number for each bit in the TE to NT data frame. It is assumed that the data for the B channels, the D channel and the Q channel has been loaded before the signal coding begins. The signal START_NEXT_FRAME starts the process and synchronises events. Bit 46 is transmitted first. Each bit within the frame may be a space (no voltage) or a positive or negative pulse. The coding rules state that no two consecutive pulses may have the same polarity. There are two exceptions. These are the two coding rule violations incorporated within the frame to provide regular timing events. The ASM chart for the state machine is presented in figures 39a, 39b, and 39c. The one-hot state encoding method was employed during the implementation. The explanation will deal with how each bit in the frame is coded and reference will be made to figure 38 and figure 39 throughout.

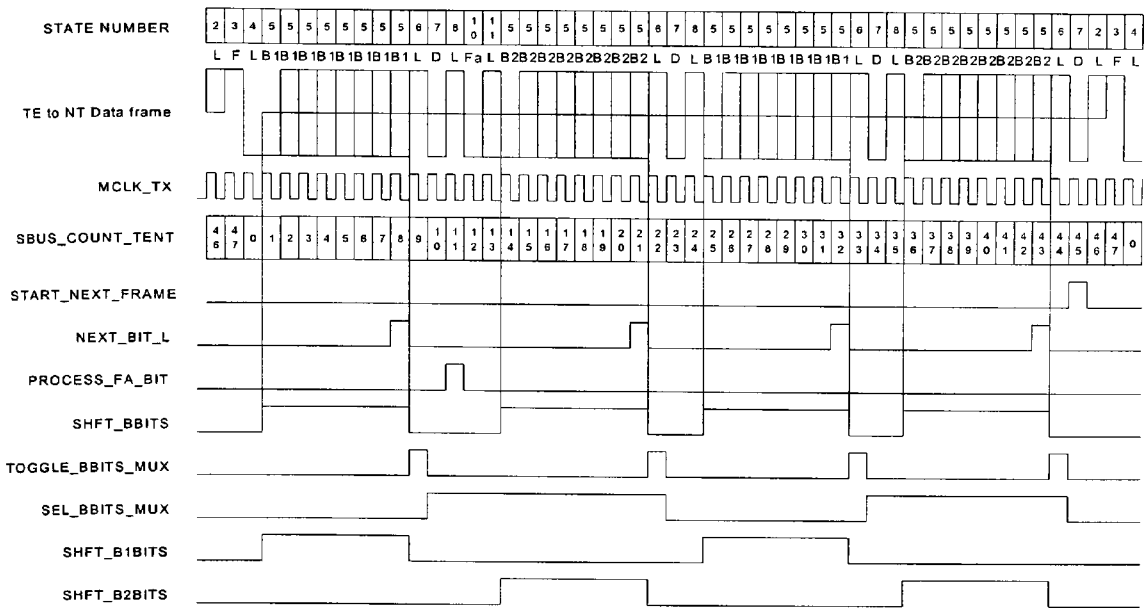


Figure 38 - The timing diagram for the INFO3_GEN block.

Bit 45: This is the fourth D channel bit. State 1 is entered when the state machine is reset. This is a wait state and the state machine moves to state 2 when the signal START_NEXT_FRAME becomes active. The signal START_NEXT_FRAME is produced when the count sequence SBUS_COUNT_TENT is 45 as shown in figure 38. In state 1 the supporting architectural blocks are reset to their default states. The coding process needs to know whether the last pulse coded was positive or negative. Two flip-flops in the INFO3_ARCH block are used to record the state of the last pulse transmitted. The state machine sets and resets these flop-flops as the process runs. The input signal, LAST_PULSE_POS, indicates that the last pulse was positive.

Bit 46: This is the balancing bit denoted as the L bit. Bit 46 is coded during state 2 shown in figure 38. A positive pulse may have occurred before this point and the LAST_PULSE_POS signal is checked to determine this. A positive pulse is produced if a positive pulse has not been generated before this point. During the

state sequence if a positive or negative pulse is generated the state of the last pulse generated is updated. It is clear from the ASM chart when this updating process takes place and this point will not dealt with further.

Bit 47: This is the Framing bit denoted as the F bit. Bit 47 is coded during state 3. This will always be coded as a positive pulse. The last pulse, either bit 46 or a symbol previous to bit 46 will have been a positive pulse also. Therefore the first coding rule violation is produced here.

Bit 0: This is a balance bit denoted as an L bit. Bit 0 is coded during state 4. This will always be coded as a negative pulse. This combination of L bit and preceding F bit is referred to as the FL pair. This is the only timing event during a basic rate ISDN frame that takes place consistently at the same place in the frame.

Bit 1 to bit 8: This is the first eight bits of the B1 channel. The B1 channel bits are coded during state 5. The B channel data is coded when the state sequence enters state 5. State 5 is executed four times as shown in the timing diagram. The architecture blocks INFO3_B1_REG both have sixteen bits of B channel data ready for coding. Each time state 5 is entered eight bits of B channel data are processed. The B channel data is extracted from the shift-registers in eight bit blocks and the appropriate data to be coded next is determined by a data multiplexing arrangement. The architecture blocks INFO3_BBITS_MUX and INFO3_BSHFTSIGNAL_MUX implement this multiplexing arrangement. The signal SEL_BBITS_MUX is set or reset using the signal TOGGLE_BBITS_MUX. The state machine produces the signal TOGGLE_BBITS_MUX, which changes the state of SEL_BBITS_MUX. State 5 processes both B1 data and B2 data alternatively. B1 data is processed if SEL_BBITS_MUX is logic low when state 5 is entered and B2 data is processed if SEL_BBITS_MUX is high. The signal SHFT_BBIT_REG is active during state 5

and this shifts B channel data serially into the state machine for processing. The INFO3_BSHFTSIGNAL_MUX routes the signal SHFT_BBIT_REG to either the register containing the B1 data or that containing the B2 data. The selected B channel data is applied to the signal NEXT_BBIT for coding by the state machine. When state 5 is entered during bits 1 through 8, it is the intention to force a violation. If the next bit on the NEXT_BBIT signal is logic 0 then the state machine produces a negative pulse. This in combination with bit 0 causes the second violation to occur. The set violation flag is set to indicate this.

Bit 9: State 6 is entered. The signal TOGGLE_BBITS_MUX is activated in order to configure the architecture to process B2 the next time state 5 is entered. If the last B data bit transmitted resulted in a negative pulse, then the L bit is set to a positive pulse here to balance the dc signal on the line.

Bit 10: State 7 is entered when bit 10 is active. State 7 processes the D channel bits. In this implementation the D channel is unused, however logic ones must be sent to disable the D channel. The signal NEXT_D is forced to logic 1 in order to achieve this. The signal START_NEXT_FRAME is checked here and will be active if the end of the frame has been reached. The sequence returns to state 2 when this is the case.

Bit 11: State 8 is entered and the L bit is processed in the same way as during bit 9. The Fa bit may be a Q bit once every four frames. The Q_BIT input is checked during state 8 and if active the next state will be state 9.

Bit 12: State 10 or state 9 may exists when the TE to NT count sequence reads 12. State 10 processes the Fa bit. If a violation has not been made already it is forced during this state. State 9 processes the Q bit if the current Fa bit is a Q bit.

Bit 13: State 11 is entered and the L bit is processed. If the last B data bit transmitted resulted in a negative pulse, then the L bit is set to a positive pulse here to balance the dc signal on the line. During state 11 the Q bits counter status is monitored. If four Q bits have been sent then the signals CLR_QBITS_RDY, CLR_QBITS_CNT are output. The signal CLR_QBITS_RDY is a synchronizing signal. This signal is sent to the downstream electronics to indicate that all the Q bits have been sent.

Bit 14 to bit 21: This is the first eight bits of the B2 channel. The B1 channel bits are coded during state 5. The B2 channel is coded in exactly the same way as B1. The procedure has already been explained for bits 1 through 8.

Bit 22: State 6 is entered. The signal TOGGLE_BBITS_MUX is activated in order to set the architecture up to process B1 the next time state 5 is entered.

Bit 23: State 7 processes the D channel bits. In this implementation the D channel is unused, however logic ones must be sent. The signal NEXT_D is forced to logic 1 in order to achieve this.

Bit 24: State 8 is entered and the same procedure as bit 11 is followed.

Bit 25 to 32: This is the second eight bits of the B1 channel. The B1 channel bits are coded during state 5. The procedure has already been explained for bits 1 through 8.

Bit 33: State 6 is entered. The signal TOGGLE_BBITS_MUX is activated in order to set the architecture up to process the second eight bits of B2 the next time state 5 is entered.

Bit 34: State 7 processes the D channel bits. In this implementation the D channel is unused, however logic ones must be sent. The signal NEXT_D is forced to logic 1 in order to achieve this.

Bit 35: State 8 is entered and the same procedure as bit 11 is followed.

Bit 36 to 43: This is the second eight bits of the B2 channel. The B1 channel bits are coded during state 5. The procedure has already been explained for bits 1 through 8.

Bit 44: State 6 is entered. The signal TOGGLE_BBITS_MUX is activated in order to set the architecture up to process the first eight bits of B1 the next time state 5 is entered.

Bit 45: State 7 processes the D channel bits. In this implementation the D channel is unused, however logic ones must be sent. The signal NEXT_D is forced to logic 1 in order to achieve this. The signal START_NEXT_FRAME goes active when the TE to NT sequence count is 45. This returns the state sequence to state 2 and the next frame of data is processed.

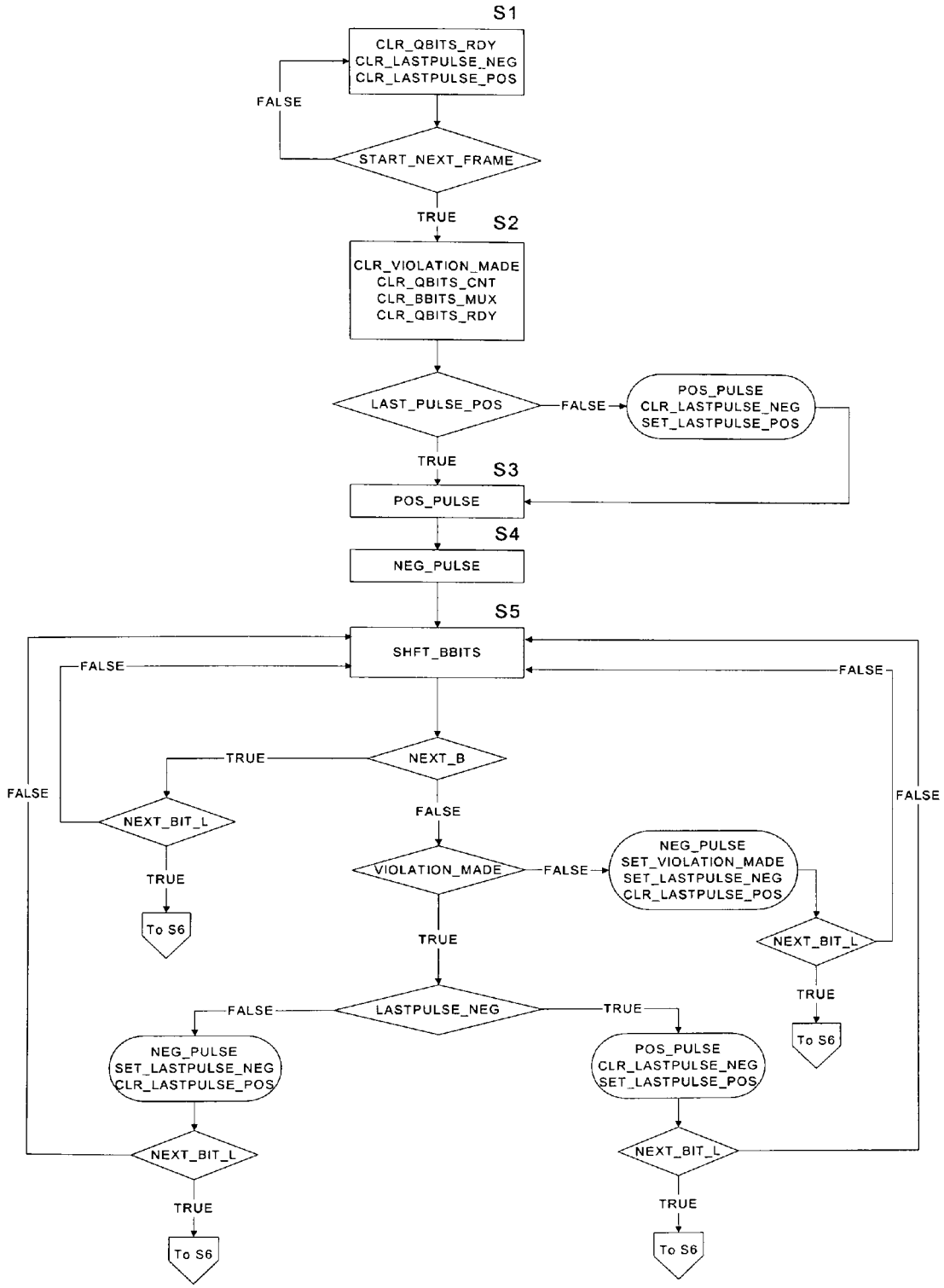


Figure 39a - The INFO3_GEN state machine.

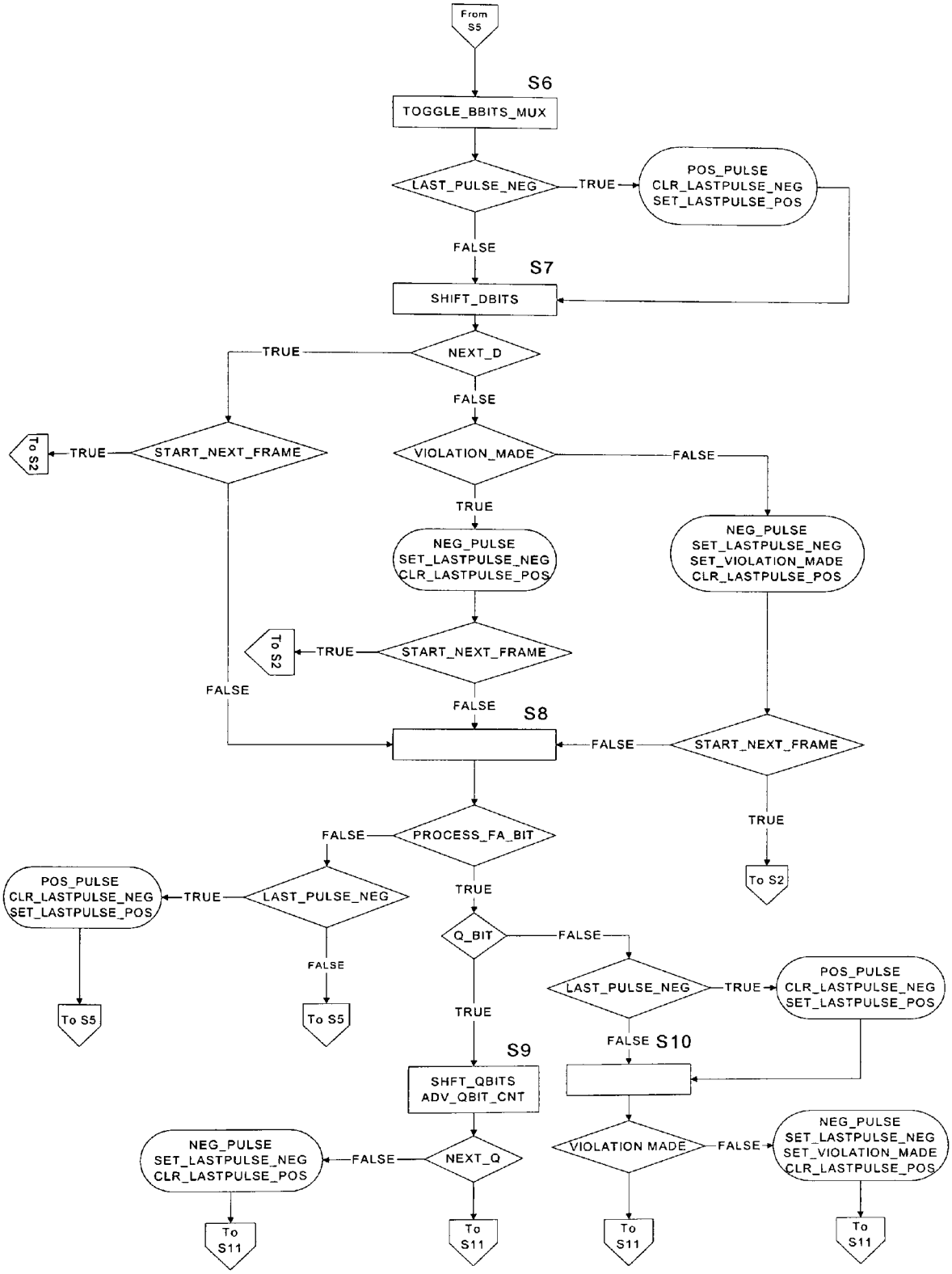


Figure 39b - The INFO3_GEN state machine.

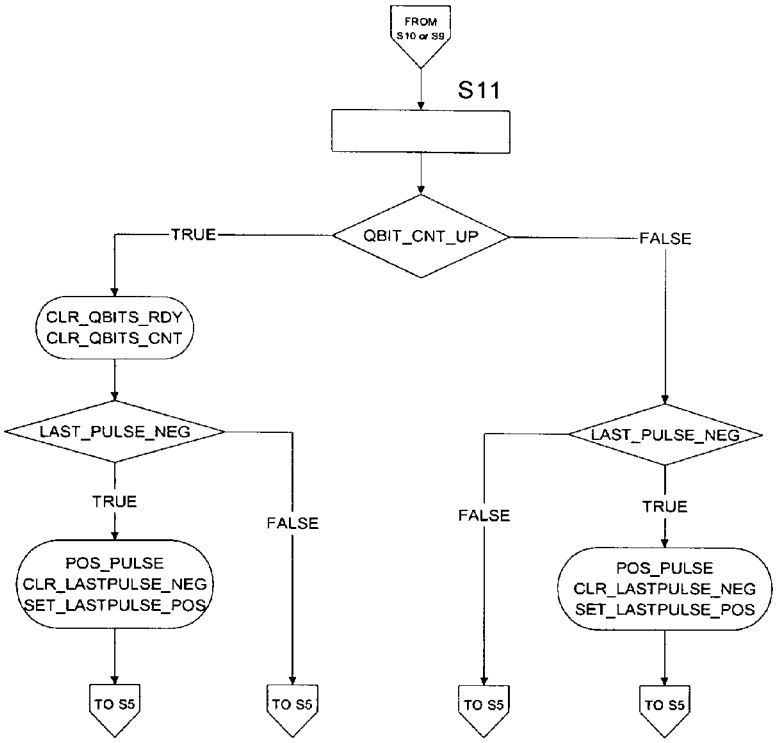


Figure 39c - The INFO3_GEN state machine.

4.5 The Loopback block.

The Loopback block is a function of the TE and performs the synchronisation with the maintenance channel. The maintenance channel is embedded within the data frames from the NT. When synchronisation with the maintenance channel is established the Loopback block sends a loopback request via the Q channel to the NT. The TE may request a loopback on the B1 channel or the B2 channel. The NT may also provide a simultaneous loopback on B1 and B2. When the NT receives a loopback request from the TE, it provides the loopback facility and sends an indication of the active loopback in the S channel to the TE. The loopback block may begin transmission of a PRBS once indication of the active loopback is received from the NT. The Loopback block can now monitor the PRBS sequence on the looped back channel and record bit errors in the received sequence.

The block diagram for the Loopback block is presented in figure 6 of appendix a. The block diagram shows the sub-blocks in the design. The MULTI_CHAN_SYNC block performs the synchronisation with the maintenance channel. The MULTI_CHAN_DECODER reads data from the S channel and decodes the S channel codes. The MULTI_CHAN_ENCODER performs the encoding of loopback requests into Q channel codes. The PRBS_TXRX block performs the PRBS generation and PRBS receiver operation. The SBUSCNT_LB_DECODE monitors the NT to TE count sequence from the SBUS_DLL block and generates the synchronisation events for the architecture. The START_NEXTFRAME_SYNC monitors the TE to NT count sequence from the SBUS_DLL block and facilitates the PRBS transmission process. The following is a description of each block within the Loopback design.

4.5.1 The SBUSCNT_LB_DECODE block.

The data on the S bus from the NT is synchronised with the sampling clock to create the signal DATA_IN. The input signal SBUSCNT_NTTE represents the count sequence used to identify each bit within a data frame from the NT. The SBUSCNT_LB_DECODE block decodes the count sequence and the following signals are produced in synchronism with the MCLK_RX clock signal.

S_BIT: This signal identifies when the S bit data is valid on the DATA_IN signal and is active when the count sequence reads 35.

FA_BIT: This signal identifies when the Fa bit data is valid on the DATA_IN signal and is active when the count sequence reads 12.

LAST_E_BIT: This signal identifies when the last E bit within the data frame is valid on the DATA_IN signal. This signal goes active when the count sequence reads 44.

M_BIT: This signal identifies when the M bit data is valid on the DATA_IN signal and is active when the count sequence reads 24.

N_BIT: This signal identifies when the N bit data is valid on the DATA_IN signal and is active when the count sequence reads 13.

START_NEXTFRAME_NTTE: This signal goes active when the count sequence reads 47.

B1_START_NEXT: This signal goes active when the count sequence reads 0 or 24. This indicates that B1 channel data is valid on the next clock cycle.

B1_END_NEXT: This signal goes active when the count sequence reads 32 or 8 and indicates the end of B1 data on the DATA_IN signal.

B2_START_NEXT: This signal goes active when the count sequence reads 13 or 35. This indicates that B2 channel data is valid on the next clock cycle.

B2_END_NEXT: This signal goes active when the count sequence reads 21 or 43 and indicates the end of B2 data on the DATA_IN signal.

4.5.2 The MULTI_CHAN_SYNC block.

The MULTI_CHAN_SYNC block performs the synchronisation process with the maintenence channel from the NT. Synchronisation with the NT must first be established before synchronisation with the maintenance channel can begin. Synchronisation with the maintenance channel is established by monitoring the data frames from the NT and hunting for the frame structure presented in table 2. The frame structure consists of twenty frames called a multi-frame. The multi-frame synchronisation process waits for frame number 1 when the Fa bit and the M bit are both binary one. This provides a datum from which to derive timing. The N bit in the NT to TE direction is always the binary opposite of the Fa bit and this is used as an extra validation check during the process. The binary values of the Fa, N and M bits are captured and evaluated at the end of each frame.

A state machine monitors the frame structure over time and hunts for the frame structure presented in table 2. Once this has been established the position of the maintenance channel can be identified. The maintenance information is transported to the TE as a four bit word every twenty frames. The SC11, SC12, SC13 and SC14 are the maintenance channel bits. The MULTI_CHAN_SYNC block is composed of a number of sub-blocks. The block diagram for the system is presented in figure 7 of appendix a. The following describes each hardware block in the design.

Frame number	NT-to-TE F_A-bit position	NT-to-TE M bit	NT-to-TE S bit
1	ONE	ONE	SC11
2	ZERO	ZERO	SC21
3	ZERO	ZERO	SC31
4	ZERO	ZERO	SC41
5	ZERO	ZERO	SC51
6	ONE	ZERO	SC12
7	ZERO	ZERO	SC22
8	ZERO	ZERO	SC32
9	ZERO	ZERO	SC42
10	ZERO	ZERO	SC52
11	ONE	ZERO	SC13
12	ZERO	ZERO	SC23
13	ZERO	ZERO	SC33
14	ZERO	ZERO	SC43
15	ZERO	ZERO	SC53
16	ONE	ZERO	SC14
17	ZERO	ZERO	SC24
18	ZERO	ZERO	SC34
19	ZERO	ZERO	SC44
20	ZERO	ZERO	SC54
1	ONE	ONE	SC11
2	ZERO	ZERO	SC21
etc.			
NOTE – S-subchannels not used by the NT1 shall be set to all binary ZEROs.			

Table 2 - The Maintenance channel structure [2].

The FAMBIT_REG block.

The FAMBIT_REG block implements a shift-register that captures the Fa, N and M bits in the data frames transmitted from the NT. A block diagram of the system is shown in figure 40. The shift-register has eleven register elements. The register elements Q9 through Q0 store the logic state of the Fa and N bits within each frame from the NT. The register element labeled MREG stores the logic state of the M bit within each frame. When twenty frames corresponding to the frame structure presented in table 2 are transmitted the registers will hold the data bits as shown in figure 40. The inputs and outputs from the block are presented next.

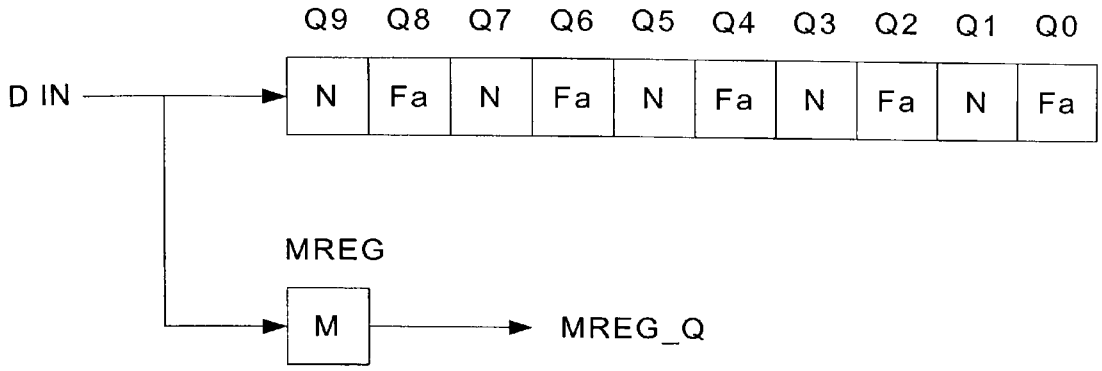


Figure 40 - The FAMBIT_REG block diagram.

The FAMBIT_REG block inputs.

CLK: This is the clock signal. This is connected to the MCLK_RX signal downstream.

RST: This is a synchronous hardware reset.

DIN: This is the data stream from the NT synchronized with the MCLK_RX clock.

FABIT: This signal is from the SBUSCNT_LB_DECODE block and enables the shift-register when the Fa bit is active on DIN. The Fa bit is then synchronously loaded into the shift-register on the next rising edge of the clock.

NBIT: This signal is from the SBUSCNT_LB_DECODE block and enables the shift-register when the N bit is active on DIN.

MBIT: This signal is from the SBUSCNT_LB_DECODE block and enables the shift-register when the M bit is active on DIN.

The FAMBIT_REG block outputs.

FA_N_M: The signal FA_N_M indicates the frame number 1 exists. The signal is derived from the shift-register when the following condition holds true.

FA_N_M = ((not Q(9)) and (Q(8)) and MREG_Q).

LAST_5FRAMES_OK: This indicates that the logic state of the Fa and N bits from the last five frames has been compliant with the frame structure in table 2. The following equation is applied to the shift-register data to produce the signal LAST_5FRAMES_OK.

LAST_5FRAMES_OK = Q(9) and (not Q(8)) and Q(7) and (not Q(6)) and Q(5) and (not Q(4)) and Q(3) and (not Q(2)) and (not Q(1)) and Q(0).

The FRAME_CNT block.

The FRAME_CNT block implements a counter with twenty states. The count sequence is represented by the signal FRAME_CNT in the timing diagram presented in figure 41. The counter is used to count data frames during the S channel synchronization process. The inputs and outputs are described in the following.

The FRAME_CNT block inputs.

CLK: This is the clock signal.

RST_ FRAMECNT: This is a synchronous hardware reset.

ADV_FRAMECNT: This advances the count sequence.

The FRAME_CNT block outputs.

SUBFRAME: This signal goes active when the count sequence is 4,9,14 and 19 and indicates when the current frame is a sub-frame.

MULTIFRAME: This signal goes active when the count sequence is 19 and indicates that twenty frames have been processed.

PROCESS_QBIT: This signal goes active when the count sequence is 4,9,14 and 19 and identifies a Q bit in the TE to NT direction. This signal is not used in the current implementation.

The 5FRAME_CNT block.

The 5FRAME_CNT implements a counter with five states. The counter is advanced to the next state after each sub-frame. The signal SUBFRAME_CNT in figure 40 represents the count sequence during the multi-frame synchronisation process. The inputs and outputs are as follows.

The 5FRAME_CNT block inputs.

CLK: This is the clock signal.

ADV_5FRAMECNT: This advances the counter.

RST_5FRAMECNT: This is a synchronous reset to the counter.

The 5FRAME_CNT block output.

FRAMECNT5_UP: When the count sequence reaches 4 this signal goes active.

The CRIT_CNT block.

The CRIT_CNT block implements two-bit binary counters and is used to count the number of consecutive times a multi-frame is identified. The count sequence runs from 0 to 3. The CRIT_CNT block is used twice (U17 and U18) in the design and this can be seen in the block diagram. U17 is used to count the number of

consecutive times a multi-frame is identified. When synchronisation is established with the maintenance channel, U18 is used to count the number of consecutive times a multi-frame is not identified. This allows the state-machine to check the persistence of a condition over a number of frames and thus a level of noise tolerance is built into the design.

The CRIT_CNT block inputs.

CLK: This is the clock signal.

ADV_CCNT: This advances the count sequence.

RST_CCNT: This resets the count sequence to 0.

The CRIT_CNT block outputs.

CCNT: This signal goes active when the count sequence is 3.

The MULTICHAN_FLAGS block.

The MULTICHAN_FLAGS block implements a set/reset flip-flop that indicates when synchronisation with the maintenance channel has been established.

The MULTICHAN_FLAGS block inputs.

CLK: This is the clock.

RST_MULTISYNC: This resets the flip-flop to logic 0.

SET_MULTISYNC: This sets the flip-flop to logic 1.

The MULTICHAN_FLAGS block output.

MULTICHAN_SYNC: Synchronisation with the maintenance channel is indicated when set to logic 1.

The MULTICHAN_CON block.

The MULTICHAN_CON block implements the state-machine controller for the maintenance channel synchronisation process. The state-machine relies on the architectural blocks already described to facilitate the process. The inputs and outputs for the state-machine have already defined in the preceding sections where the architecture within the MULTI_CHAN_SYNC block was described. The timing diagram in figure 41 will be used to describe the state-machine operation. The timing diagram shows the state sequence as the frames progress through time. The timing diagram is not to scale, however it serves to describe the operation of the state-machine over one multi-frame. Each box within the frame sequence is numbered with its frame number. The frame sequence shows each frame on the diagram and indicates the bits that are significant in establishing synchronisation. For example, frame number 1 indicates that the M bit, Fa bit, and the N bit are active. The state machine monitors the logic state of these bits and hunts for a regular occurrence of the frame structure detailed in table 2. Once this has been identified the position of the SC1 bits can be indicated by the signal FRAME_NO5, also called SUBFRAME in the higher levels of hierarchy.

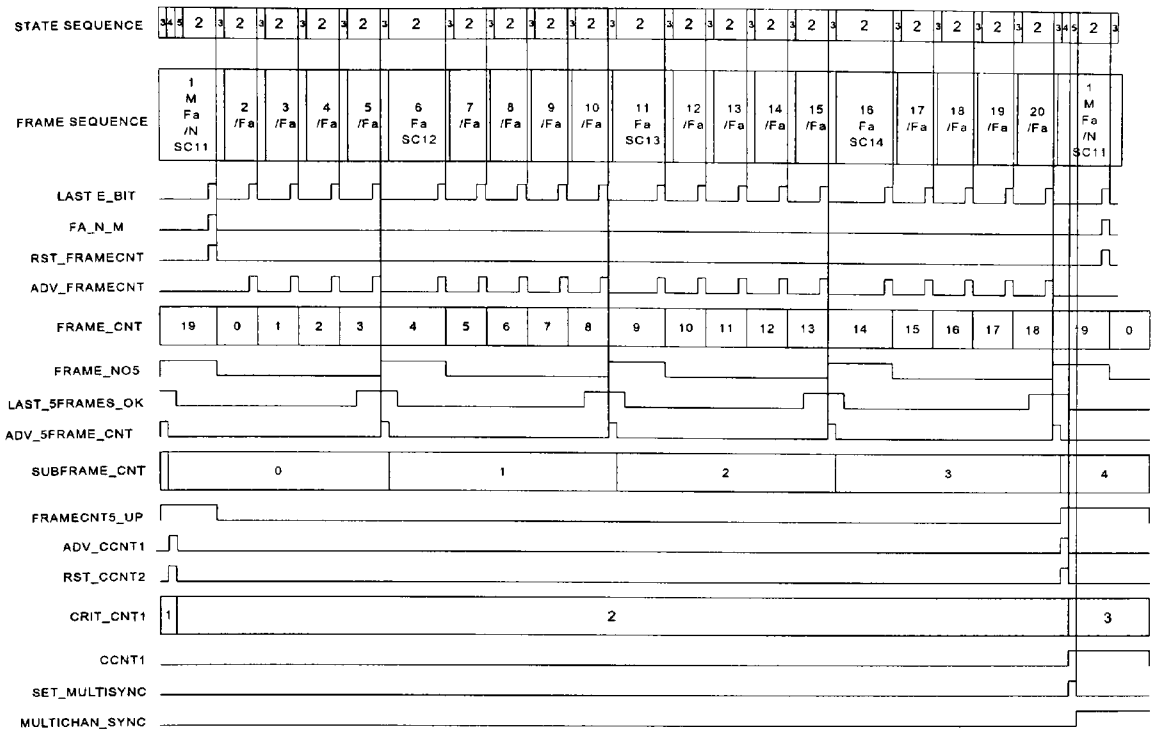


Figure 41 - The MULTICHAN_CON timing diagram.

The state machine has five states and the following describes each of these states in turn. The timing diagram in figure 41 was used at the design stage to derive the state machine algorithm. The ASM chart is shown in figure 42a and 42b. The one-hot state encoding method was employed at the implementation stage.

State 1 (S1): This is the default state that is entered after a synchronous reset. Here the architectural blocks are reset to their default states.

State 2 (S2): This state waits until the signal LAST_EBIT goes active. This ensures that the FAMBIT_REG block has read the state of the Fa bit, N bit and M bit from each frame before checking results. When LAST_EBIT goes active the signal FA_N_M is checked. If this signal is active high then frame number 1 in the multi-frame has been identified. The frame count sequence (FRAME_CNT) is reset to 0.

If the FA_N_M is low when LAST_EBIT goes active the frame count sequence is advanced with ADV_FRAMECNT.

State 3 (S3): When the signal FRAME_NO5 goes active, indicating that a sub-frame structure has been processed, the Fa and N bits of the last five frames are checked. A sub-frame consists of five frames. The FAMBIT_REG produces logic 1 on the signal LAST_5FRAMES_OK when the sampled Fa and N bits conform to the correct sub-frame structure. On each sub-frame the sub-frame count sequence (SUBFRAME_CNT) is advanced only if the sub-frame structure is correct. The state sequence moves to state 4 if LAST_5FRAMES_OK is logic 0 during a sub-frame indication or when FRAME_NO20 is high indicating a multi-frame has passed.

State 4 (S4): During state 4 the CRIT_CNT blocks U17 and U18 are updated. The counters each have four states. Each time four good sub-frames are identified within a multi-frame the count sequence for U17 (CRIT_CNT1 in the timing diagram) is advanced and the count sequence for U18 is reset. The opposite is true each time a bad sub-frame is identified within a multi-frame. If the state-machine can consistently identify four good multi-frames conforming to the structure in table 1, then U17 will advance through the four states. When U17 reaches its fourth state the signal CCNT1 goes active.

State 5 (S5): The counters U17 and U18 are checked in state 5. When the signal CCNT1 from U17 is true, this indicates that a consistent multi-frame structure can be identified. In this case the signal SET_MULTISYNC sets the multi-frame synchronisation flag. If multi-frame synchronisation is lost then the signal CCNT2 from the counter U18 will go active. This will reset the multi-frame synchronisation flag. The state sequence leaves state 5 and returns to state 2 to continue the process.

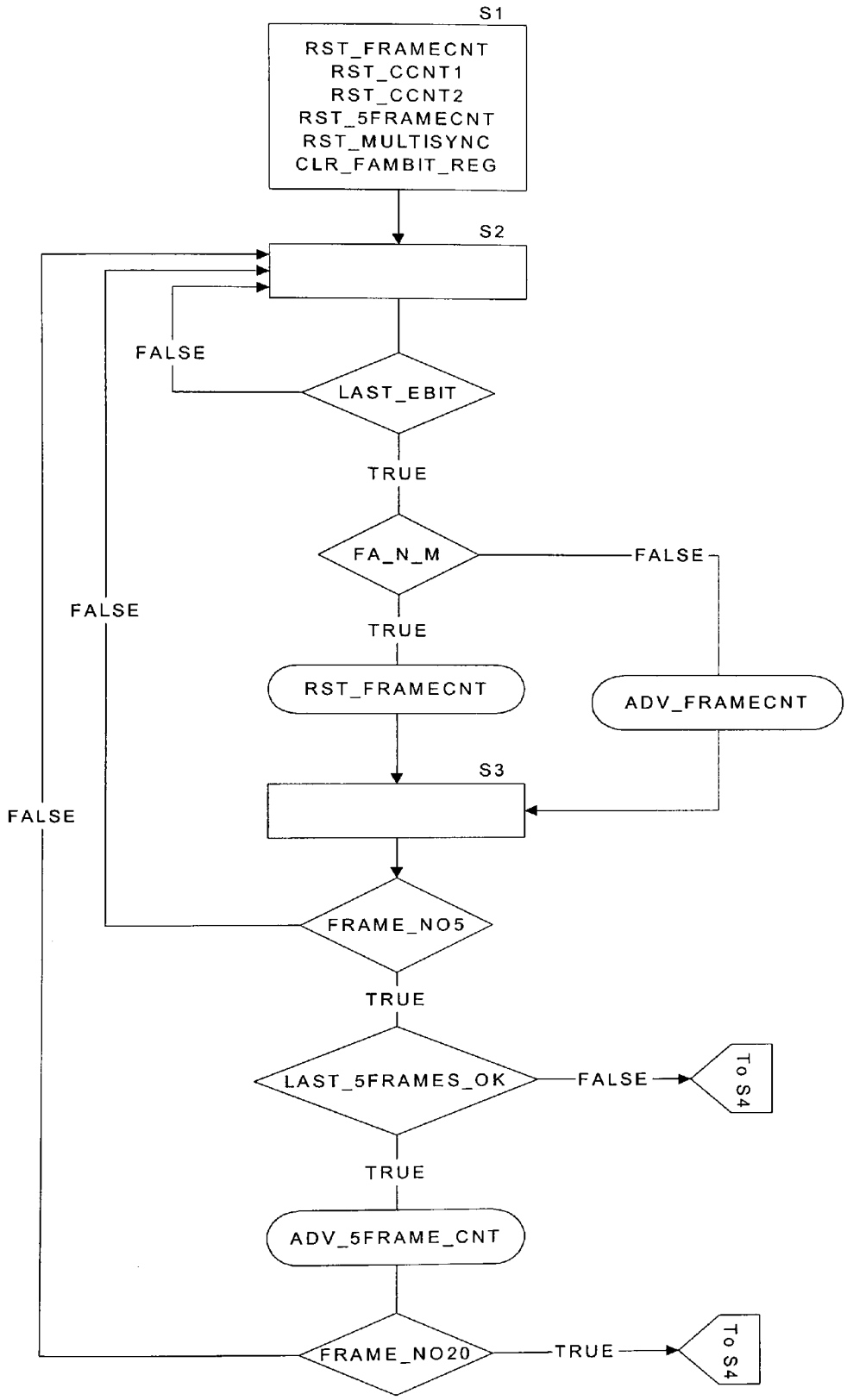


Figure 42a - The MULTICHAN_CON ASM chart.

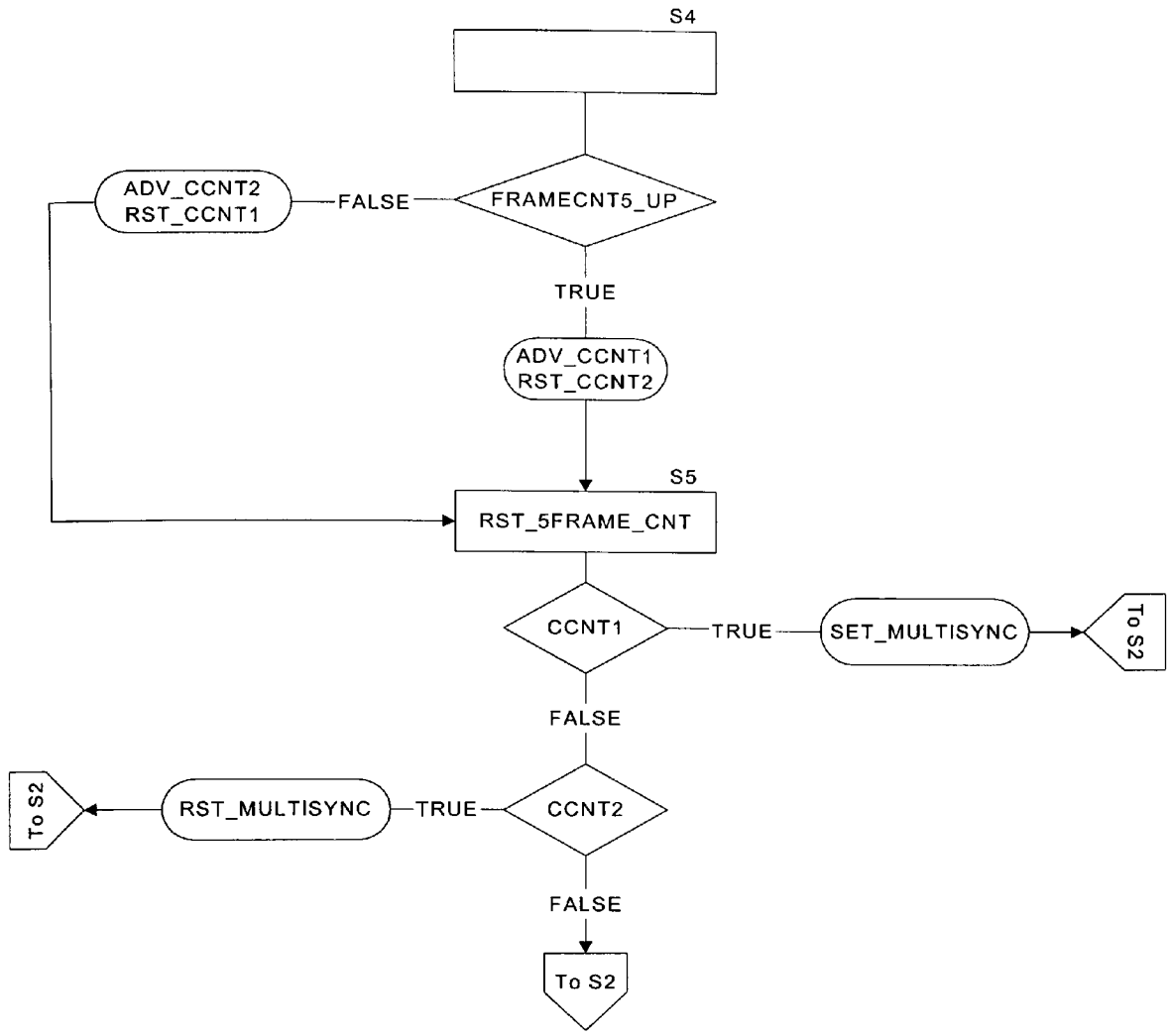


Figure 42b - The MULTICHAN_CON ASM chart.

4.5.3 The MULTICHAN_DECODER block.

The block diagram for the MULTICHAN_DECODER block is presented in figure 8 of appendix a. The design consists of three sub-blocks, the SBITS_REG, SBUS_LATCH, and the SCHAN_DECODER. When synchronisation with the maintenance channel is achieved the MULTICHAN_DECODER block latches the logic state of the SC1 bits and decodes the SC1 channel into one of ten status signals. The NT sends the current status on the SC1 channel continuously until a change of status exists. The objective of the MULTICHAN_DECODER block is to capture the SC1 bits from each multi-frame and produce an up to date status indication. This implementation is concerned with the loopback status only. The facilities to indicate the other status conditions have been provided but not tested. The sub-blocks in the design are explained next.

The SBIT_REG block.

The SC1 bits are shifted into a serial in to parallel out shift-register implemented by the SBITS_REG block. The SUBFRAME signal from the MULTI_CHAN_SYNC block is connected to the LATCH_SC1BIT signal to indicate the existence of a frame carrying an SC1 bit. The signal, LATCH_SC1BIT enables the SBITS_REG and the next SBIT signal from the SBUSCNT_LB_DECODE block registers the logic state of the S bit on the DIN line.

The SBUS_LATCH.

The signal MULTIFRAME from the MULTI_CHAN_SYNC indicates that all four SC1 bits are available on the SC bus. The signal MULTIFRAME then latches the four bits into the SBUS LATCH block. The SBUS_LATCH block implements a

parallel latch. The contents of the SC bus are registered to produce the bus SC_LATCH. The status of the SC1 bits will be updated once every multi-frame. The SC_LATCH ensures that the SC1 bits remain stable while the decoding operation is being performed.

SCHAN_DECODER.

The SCHAN_DECODER implements a logic decoder according to table 3. The signal LB1_L indicates that a loopback has been put on the B1 channel. The signal LB2_L indicates that a loopback has been put on the B2 channel. The signal LB1ANDLB2_L indicates that a loopback has been put on the B1 and the B2 channel. The signal IDLE_L indicates that no loopback condition exists.

Status Indication	SC14	SC13	SC12	SC11
LB1_L	1	0	1	1
LB2_L	1	1	0	1
LB1ANDLB2_L	1	0	0	1
IDLE_L	0	0	0	0

Table 3 - The SC1 channel status indication signals.

4.5.4 The MULTI_CHAN_ENCODER block.

The MULTI_CHAN_ENCODER block diagram is presented in figure 9 of appendix a. The objective of the block is to provide the Q bits for the Q channel in the TE to NT direction of data transmission. The Q bits are sent to the INFO3_GEN

block for processing. It is the responsibility of the MULTI_CHAN_ENCODER block to update the Q bits in the INFO_GEN block every multi-frame. The MULTIFRAME signal from the MULTI_CHAN_SYNC block is monitored and an edge-detector generates the signal LATCH_QBITS on the positive going edge of the MULTIFRAME signal. The signal LATCH_QBITS is sent to the INFO_GEN block and this latches the Q bits for processing during the next multi-frame. The action occurs synchronously with the MCLK_TX clock signal. The MULTI_CHAN_ENCODER block has one sub-block called the QBITS_ENCODER block.

The QBITS_ENCODER block.

The QBITS_ENCODER block produces the Q channel codes to request loopback facilities from the NT. The Q bit codes are produced in accordance with those defined in recommendation I.430. The inputs to the block are defined as follows. Only one input can be active at any one time. The resulting Q channel loop back request codes are shown in table 4.

LOOPBACK_B1: When this is set to logic one the Q channel code to request a loop back on channel B1 is generated.

LOOPBACK_B2: When this is set to logic one the Q channel code to request a loop back on channel B2 is generated.

LOOPBACK_B1B2: When this is set to logic one the Q channel code to request a loop back on channel B1 and B2 is generated.

Loopback type	Q4	Q3	Q2	Q1
LOOPBACK_B1	1	1	1	0
LOOPBACK_B2	1	1	0	1
LOOPBACK_B1B2	1	1	0	0
No Loop back	0	0	0	0

Table 4 - The Q channel request codes.

4.5.5 The PRBS_TXRX block.

The PRBS_TXRX inserts the PRBS sequence into the S bus data frames toward the NT and extracts the PRBS sequence from the data frames that are looped back by the NT toward the TE. The extracted PRBS sequence is monitored for errors. The block diagram for the PRBS_TXRX block can be seen in figure 10 of appendix a. The design can be partitioned into PRBS transmit circuitry and PRBS receiver circuitry. The block diagram for the system is entitled PRBS_TXRX and is shown in appendix a. The PRBS_TXRX block consists of a number of sub-blocks. The sub-blocks employed to facilitate PRBS transmission are the BBUFFER blocks, the MUX_PRBSOR5V, the PRBS_TX block, the PRBSBIT_CNT block, the LB_SIGNAL_DECODE block, and the PRBSTX_CON block. These blocks are all clocked with the CLK_TX clock signal. The CLK_TX signal is connected to MCLK_TX on the top level of the design hierarchy. The sub-blocks involved in the

PRBS extraction and error monitoring are the PRBS_ERR_CNT block, the PRBS_RX block, PRBSRX_CON block and the MUX_B1ORB2 block. These blocks are clocked with the CLK_RX clock signal, which is connected to the MCLK_RX in the top level of the design hierarchy.

The PRBS transmission circuitry.

PRBS transmission begins when synchronisation with the maintenance channel is achieved and a loopback indication is received from the NT. PRBS will be inserted into the looped back channel. A basic rate ISDN data frame contains sixteen bits of data for each B channel. The PRBS_TX block generates the PRBS data. The PRBSTX_CON block implements a state machine that controls the process. The timing diagram for the process is shown in figure 43. The count sequence SBUSCNT_TENT is decoded to produce the signal START_NEXT_FAME_TENT. When the signal START_NEXT_FAME_TENT goes active the state machine generates sixteen bits of PRBS data. The sixteen bits of PRBS data is multiplexed through to a serial in parallel out buffer when the signal EN_PRBS is active.

Two buffer circuits (BBUFFER block) are employed, one for each B channel. U22 services the B1 channel and U23 services the B2 channel. The multiplexers implemented by the MUX_PRBSOR5V blocks route either the generated PRBS sequence or logic one to each buffer. The LB_SIGNAL_DECODE block provides the select signal for the multiplexers. If the loopback is on B1 the multiplexers are configured so that PRBS data is loaded into the buffer servicing the B1 channel. Sixteen logic ones are loaded into the buffer for B2. This ensures that the INFO_GEN_TENT block sends AMI spaces in the idle B channel. When the signal PRBS_16BITS_SENT goes active, the respective buffers will have been

loaded with data ready for latching by the INFOGEN_TENT block. The INFOGEN_TENT block stores the data from both buffers when the SBUSCNT_TENT count sequence reaches 46. While the INFOGEN_TENT block is transmitting sixteen bits of PRBS data, the LOOPBACK block is preparing another sixteen bits of PRBS to be transmitted during the next frame. The hardware sub-blocks will be explained next.

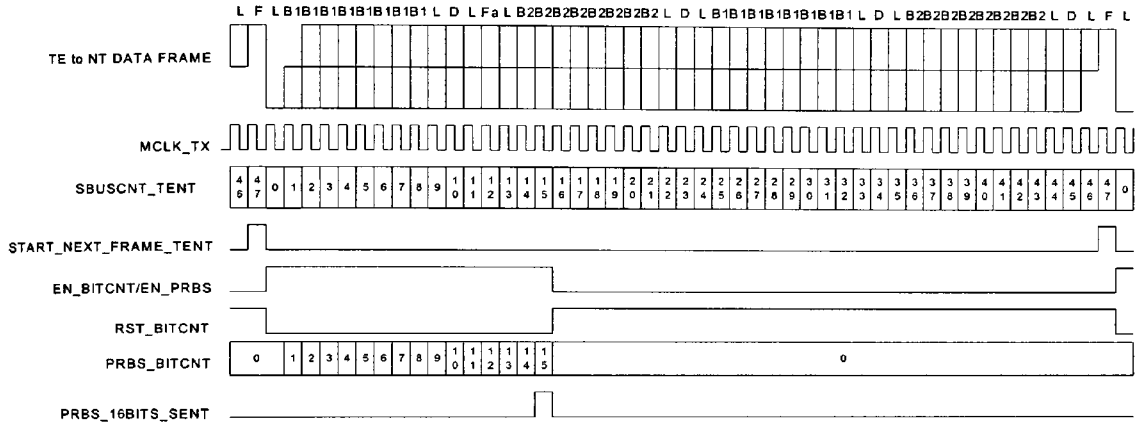


Figure 43 - The timing diagram for the PRBS transmitter.

The BBUFFER block.

The BBUFFER block implements a serial in to parallel out shift register. Two BBUFFER blocks are employed in the design. The inputs and outputs for the block are presented in the following.

DATA_IN: This is the serial data input.

SHFT: This input enables the shift-register.

SET: This input sets each register within the shift-register to logic 1.

CLK: This is the clock signal.

BDATA: This is the sixteen-bit parallel output bus containing the PRBS data.

The MUX_PRBSOR5V block.

The MUX_PRBSOR5V block implements a multiplexer. The input and outputs presented next.

PRBS: This is the PRBS input signal.

SEL: This is the multiplexer select signal. If SEL is high the PRBS input is routed to the output. If SEL is low then logic 1 is routed to the output.

DATA_OUT: This is the output signal.

The PRBS_TX block.

The PRBS_TX block implements the pseudo-random binary sequence generator. The PRBS sequence generator conforms to that described in 5.2 of the O.150 recommendation for specifications of measuring equipment defined by the ITU-T. The PRBS sequence is generated by an eleven-stage shift-register whose ninth and eleventh stage are applied to an exclusive or gate. The exclusive or gate result is fed back to the input of the first stage. The PRBS generator can never be allowed to enter a state where all register outputs are logic zero, as generation of PRBS will stop. When the generator is reset, the registers are set to a pre-defined value. In this case the first register stage is set to logic 1 while the other ten are reset to logic 0. The following describes the inputs and outputs for the block.

EN_PRBSTX: This input signal enables operation of the PRBS generator.

RESYNC: This input sets the registers to their default states and restarts the generator.

CLK: This is the clock input.

PRBS_OUT: This is the PRBS output.

The PRBSBIT_CNT block.

The PRBSBIT_CNT block implements a sixteen state counter. The state-machine controller monitors the counter during the PRBS generation procedure to limit the number of generated bits to sixteen. RST_BIT_CNT: This input synchronously resets the counter.

EN_BIT_CNT: This input enables counter operation.

CLK: This is the clock input.

PRBS16BITS_SENT: This output signal goes high when the counter sequence reaches the sixteenth state.

The LB_SIGNAL_DECODE block.

The signals LB1_I, LB2_I and LB1ANDLB2_I from the MULTI_CHAN_DECODER block indicate which loopback is active. The LB_SIGNAL_DECODE block decodes the loopback indicator signals and sets either SEL_B1 or SEL_B2 high. The following describes the inputs and outputs of the block.

LB1_I: Indicates a loopback on the B1 channel.

LB2_I: Indicates a loopback on the B2 channel.

LB1ANDLB2_I: Indicates a loopback on the B1 and the B2 channel. The current implementation limits the loopback to either B1 or B2 so this signal will not go active.

SEL_B1: This output is set high if LB1_I is active. When SEL_B1 is active the PRBS sequence is routed through to buffer U22 and logic 1 is routed through to buffer U23.

SEL_B2: This output is set high if LB2_I is active. When SEL_B2 is active the PRBS sequence is routed through to buffer U23 and logic 1 is routed through to buffer U22.

The PRBSTX_CON block.

The PRBSTX_CON block implements the state-machine controller for the PRBS transmission circuitry. The state machine executes once every frame. The ASM chart for the state-machine is presented in figure 44. The one-hot state machine encoding method was employed during implementation. The following describes the states in the process. The state machine is enabled when the signal MULTICHAN_SYNC and either one of the signals LB1_I, LB2_I or LB1ANDLB2_I is active.

State S1: This is the default state and is entered when the state-machine is reset.

State S2: This is a wait state. When the signal START_NEXT_FRAME goes active the sequence moves to state S3. The START_NEXT_FRAME signal is decoded from the SBUSCNT_TENT count sequence and goes active when SBUSCNT_TENT is 47. During state S2 the PRBSBIT_CNT block is held in reset.

State S3: This state lasts for sixteen clock cycles. During state S3 sixteen bits of PRBS are generated and stored in either buffer U22 or buffer U23. State S3 is left when the signal PRBS16BITS_SENT goes active. The inputs and outputs for the PRBSTX_CON block are presented next.

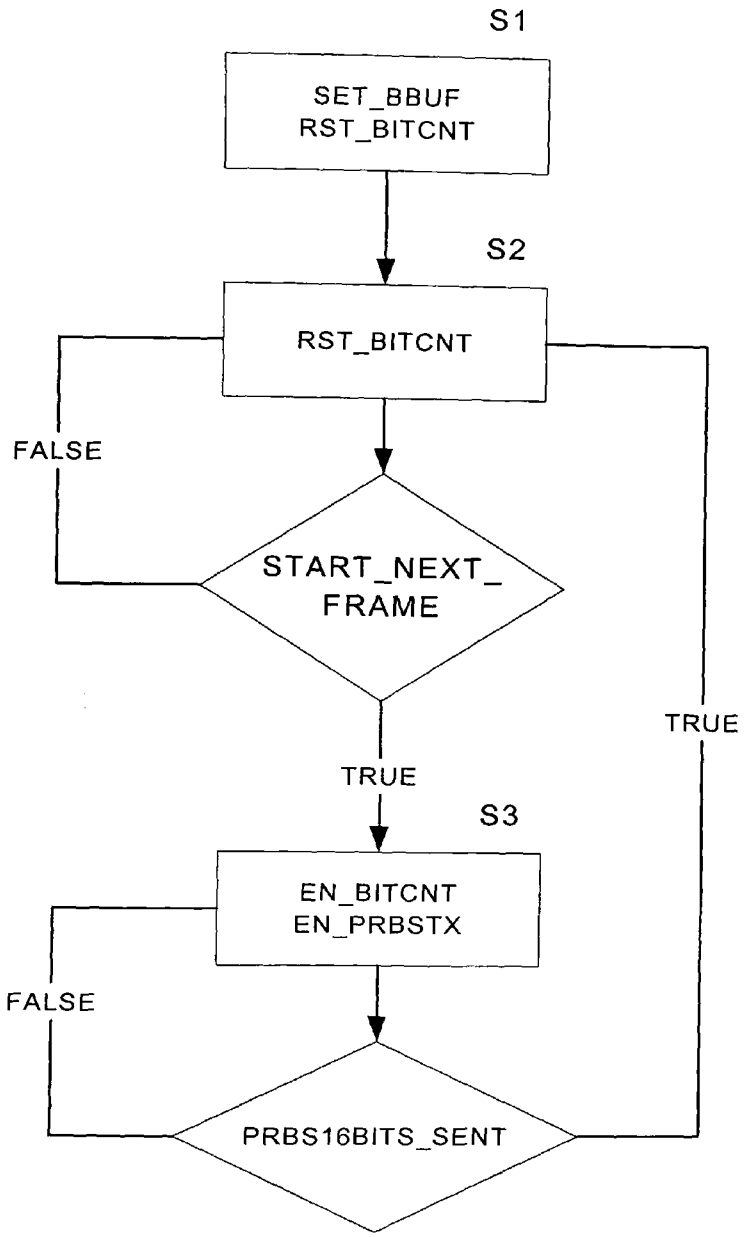


Figure 44 - The ASM chart for the PRBSTX_CON block.

The PRBS receiver circuitry.

The PRBS receiver circuitry monitors the count sequence (SBUS_NTTE) from the SBUS_DLL block. If a loopback is active on B1 then the B1 channel bits are

extracted and routed to the PRBS receiver implemented by the PRBS_RX block. The same is true in the case of a loopback on the B2 channel. The timing diagram in figure 45 shows the process for both the B1 and B2 channel. The count sequence SBUSCNT_NTTE is decoded to produce the signal B1_START_NEXT and B2_START_NEXT. These signals identify the B1 or B2 channel data within the data frame from the NT. In the same way the signals B1_END_NEXT and B2_END_NEXT identify when the B1 or B2 channels end. The PRBSRX_CON block generates the signal EN_PRBSRX. EN_PRBSRX enables the PRBS receiver operation and is derived from the before mentioned signals. The signal EN_PRBSRX is represented on the timing diagram for both the B1 and B2 channel. The signal was presented this way to emphasise the difference between the B1 channel and the B2 channel.

This implementation supports either a loopback on B1 or B2 but not both at the same time. However, the modular top-down design technique employed to realise the LOOPBACK block would support the addition of a second PRBS receiver to facilitate a loopback on both B1 and B2. The MULTI_CHAN_DECODER block identifies which loopback is active. If a loopback is active on B1 then the signal EN_PRBSRX enables the extraction of data from B1. If a loopback is active in B2 then EN_PRBSRX enables extraction of data from the B2 channel. It is necessary to synchronise the PRBS receiver with the transmitted sequence. The PRBS receiver is loaded or seeded with the received PRBS sequence. Once seeding has been completed the PRBS receiver should begin to generate the same sequence as that of the transmitter. The locally generated sequence in the PRBS receiver can now be compared to the sequence extracted from the received data frames. If the PRBS receiver senses a difference between the

sequences an error signal is generated and the error counter (the PRBS_ERR_CNT block) is advanced. The error count can be sampled over time by an external microcontroller to establish the bit error rate.

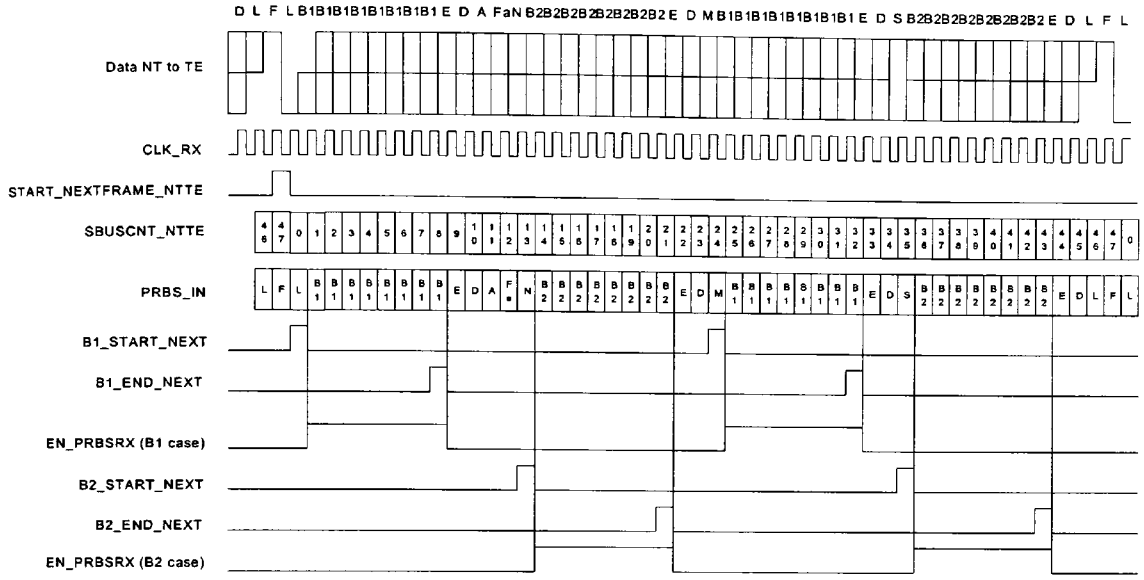


Figure 45 - The timing diagram for the PRBS receiver.

The PRBS_ERR_CNT block.

The PRBS error counter is advanced each time an error is detected by the PRBS_RX block. The block inputs and outputs are described below. The PRBS error counter sequence has 256 states.

PRBS_ERR: This input advances the PRBS error counter sequence.

EN_PRBSRX: This signal enables the counter operation.

PRBS_RST: This input resets the PRBS error counter to zero.

PRBS_ERROR: This is the eight-bit counter output. It represents an error count from 0 to 255.

The PRBS_RX block.

The PRBS_RX block implements the PRBS receiver and error monitor circuit. The PRBS receiver is composed of a PRBS generator, implemented in exactly the same way as that employed by the transmitter, a bit counter, a multiplexer, and a comparator. Figure 46 presents a simplified block diagram for the PRBS receiver. The EN_PRBS signal is connected to the clock enable of every register in the design. When EN_PRBS is high, circuit operation is enabled. The clock and the reset signal have not been shown so as to reduce the complexity of the diagram. All the registers in the design are driven with the same clock. The circuit has two stages of operation, the PRBS synchronisation stage and the PRBS error monitoring stage. The PRBS synchronisation stage is entered when a reset signal is applied to the design. When the PRBS_RX block is reset, all the registers in the PRBS generator block are reset to zero. The signal PATTERN_OK is reset low configuring the multiplexer to route the PRBS_IN signal to the MUX_OUT port. The PRBS signal from the NT can now be shifted into the PRBS generator.

The Bit counter counts the clock cycles before setting the signal PATTERN_OK high. In this application thirty-three bits of PRBS from the NT are shifted into the local PRBS generator before PATTERN_OK is set high. The PRBS error monitoring stage is entered here. At this stage the PRBS generator should have been loaded with a valid pattern. When PATTERN_OK is high the multiplexer now routes the locally generated PRBS pattern on signal LOCAL_PRBS through to MUX_OUT. The locally generated PRBS pattern should now be the same as the PRBS pattern from the NT. The comparator compares the locally generated pattern with that received on PRBS_IN on a bit by bit basis. If a difference is identified the

signal PRBS_ERR is set high. Each bit error produces a high on the PRBS_ERR signal for one clock cycle. The inputs and outputs for the PRBS_RX block are presented in the following.

PRBS_IN: This is the PRBS sequence from the NT.

EN_PRBSRX: This is the clock enable signal for the design.

RST_PRBS: This is a synchronous reset to the design.

CLK: This is the clock signal.

PRBS_ERR: This output indicates a bit error.

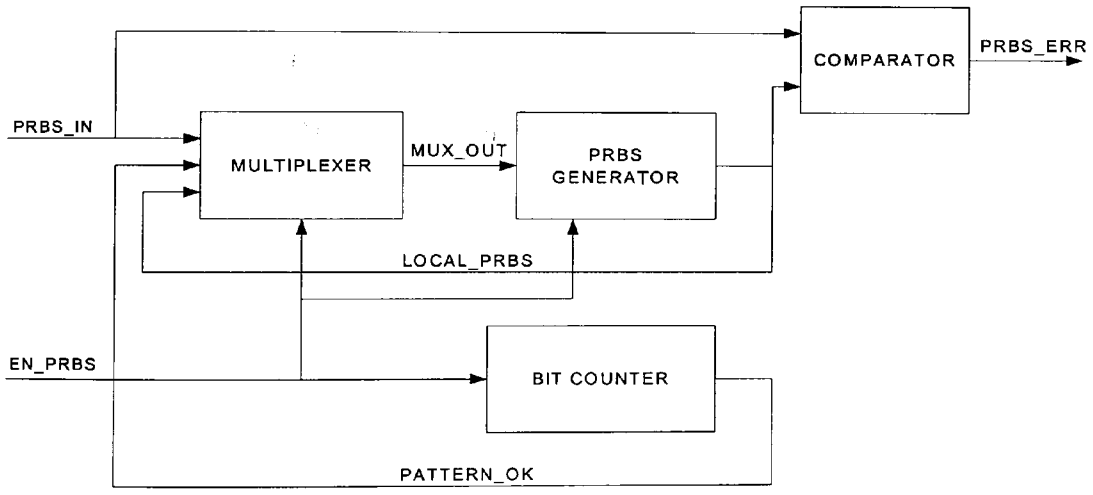


Figure 46 - The PRBS receiver block diagram.

The MUX_B1ORB2 block.

The MUX_B1ORB2 block decides which loopback is active and then routes the appropriate signals to the state-machine. The input signals B1_START_NEXT, B1_END_NEXT, B2_START_NEXT, B2_END_NEXT are routed through to the output signals B_START_NEXT, and B_END_NEXT according to the table 5.

Active Loopback	B_START_NEXT	B_END_NEXT
LB1_I	B1_START_NEXT	B1_END_NEXT
LB2_I	B2_START_NEXT	B2_END_NEXT
LB1ANDLB2_I	0	0

Table 5 - The MUX_B1ORB2 truth table.

The PRBSRX_CON block.

The PRBSRX_CON block implements the state-machine controller for the design. The ASM chart for the process is shown in figure 47. The one-hot state encoding method was used during implementation. The state-machine is enabled when synchronisation with the maintenance channel is established and a loopback is active. The following is a description of each state.

State S1 and state S2: These are wait states to allow the PRBS sequence to propagate through the NT. The state-machine waits for two frames to pass before enabling the PRBS receiver circuit. The START_NEXT_FRAME signal is decoded from the SBUSCNT_NTTE count sequence.

State S3: This state waits for the B_START_NEXT signal to go active. This identifies when to start the PRBS extraction from the S bus.

State S4: During state S4 the signal EN_PRBSRX enables PRBS receiver operation. This will continue until the signal B_END_NEXT goes active indicating the end of the B channel data on the S bus. The state sequence moves to S3 and the process continues.

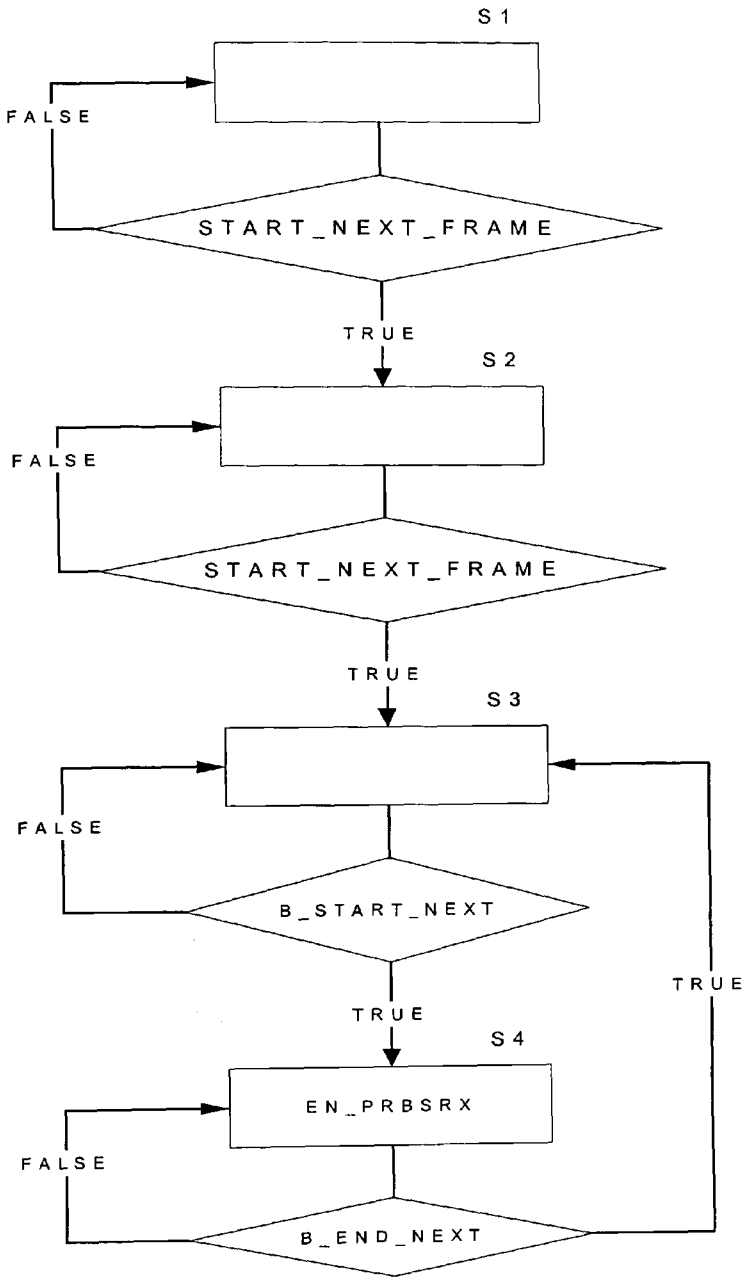


Figure 47 - The ASM chart for PRBSRX_CON block.

5 The FPGA firmware design for the network terminator.

The NT design focuses on the functionality required to facilitate test and validation of the TE circuit. The NT is the master in the Basic-rate ISDN topology and generates data frames with synchronisation data toward to TE. It is the responsibility of the TE to synchronise to these data frames. The layer 1 components of an NT design are therefore relatively straightforward. The NT generates an INFO4 signal in the NT to TE direction. The S channel is supported and the Q channel from the TE is monitored. The Q channel carries the loopback requests toward the NT. When the NT identifies a request for a loopback on either the B1 or B2 channel, it provides the loopback and sends a loopback indication in the S channel toward the TE. The loopback function in the NT buffers the B1 or B2 channel data from the TE and transmits it back to the TE in the next data frame. The NT will continue to provide the loopback while the loopback request is present in the Q channel.

The timing diagram in figure 48a shows the relationship between the data transmitted to the TE and that received from the TE. The clock signal MCLK_TX is used to clock the transmit circuitry. The count sequence SBUS_CNT_NTTE is used to identify the data bits in the NT to TE direction, labelled Data NT to TE on the diagram. The TE will synchronise to this and generate data in the TE to NT direction so that a two-bit offset exists between the two directions of data transmission. The count sequence SBUS_CNT_NTTE is decoded to produce the L_BIT signal. This signal is used to synchronise the count sequence, SBUS_CNT_TENT. The count sequence, SBUS_CNT_TENT can be decoded to identify where the B channel data bits are located within the data frames from the

TE. On the timing diagram presented in figure 48b the signal SHIFTB2_DATA is used to extract B channel data. The SHFTB2_DATA signal is derived from the START_B2 and END_B2 signals. The B1 channel is extracted in the same way. START_B2A and START_B2B identify where the B2 channel starts. Data extraction can proceed until the signals END_B2A and END_B2B identify when to stop the process.

The same procedure is used to extract data from the B1 channel. Once the data is extracted, it is stored and sent back to the TE in the next frame. The NT defines the multi-frame structure for the maintenance channel. The TE will synchronise to the multi-frame structure and the NT may extract the Q bits from the TE data frames at the appropriate times. The hardware design for the system will be presented in the following sections. The firmware block diagram is shown in figure 1 of appendix b. The block diagram partitions the NT design into the following blocks, the CLK_DIV_TEST block, the QBITS_RX block, the SBITS_TX block, the NT_LOOPBACK block and the INFOGEN_TETONT block.

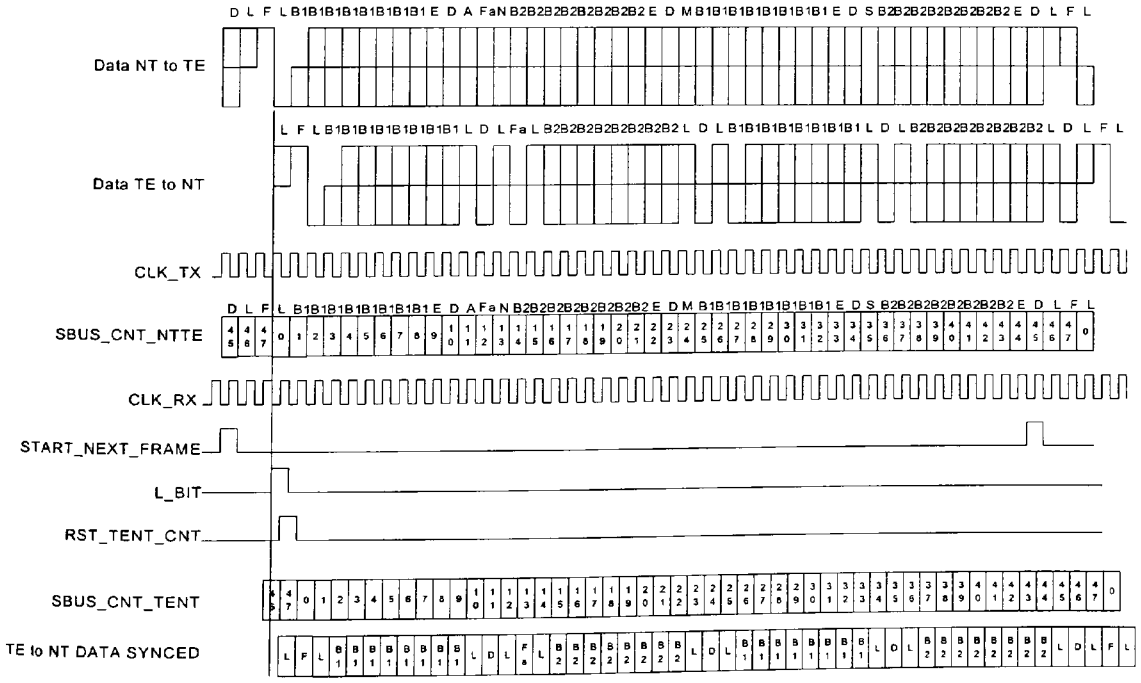


Figure 48a - Timing diagram for the NT loopback circuit.

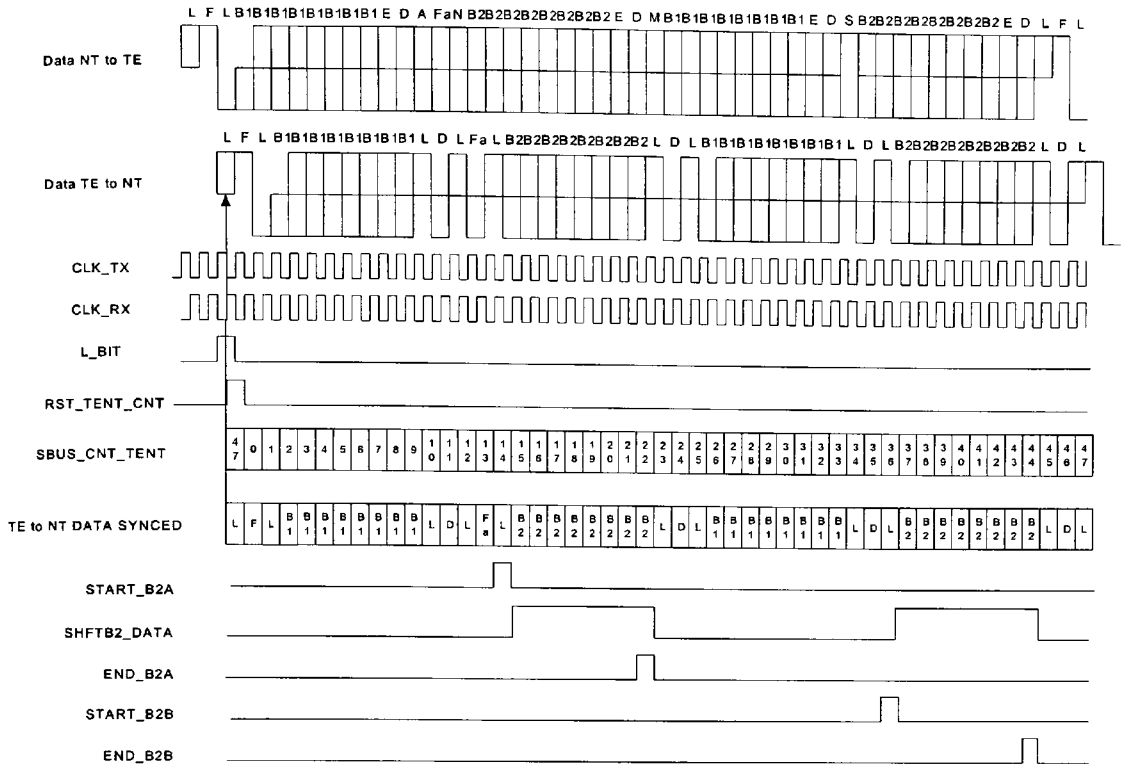


Figure 48b - The timing diagram for the NT loopback circuit.

5.1 The CLK_DIV_TEST block.

The CLK_DIV_TEST block generates transmit and receive clocks for both directions of data transmission. A count sequence is generated to identify the S bus symbols in each direction of data transmission. The timing diagram in figure 48a defines the relationship between count sequence for each direction of data transmission and the symbols on the line. The following describes the inputs and outputs for the system.

RST: This is a synchronous reset to the system.

C15_36MHZ: This input is wired to the 15.36MHz crystal frequency.

CLK_RX: This clock output has a frequency of 192kHz at 50% duty cycle and is used to clock the receiver circuitry that extracts data from the S bus.

CLK_TX: This clock output has a frequency of 192kHz at 50% duty cycle and is used to clock the transmit circuitry that inserts data into the S bus toward the TE. This is the inverse of the CLK_RX signal.

SBUS_CNT_TENT: This count sequence is broadcast to the receiver circuits and identifies the individual bits within the data frame in the TE to NT direction.

SBUS_CNT_NTTE: This count sequence is broadcast to the transmit circuits and identifies the individual bits within the data frame in the NT to TE direction.

The CLK_DIV_TEST block is composed of the CLK_DIV, DIV_BY_2, SBUS_BIT_CNT and TEST_DECODER sub blocks. The block diagram for the CLK_DIV_TEST is shown in figure 2 of appendix b. The CLK_DIV and DIV_BY_2 blocks generate the clocks and the timing diagram for the process is shown in figure 49. The 15.36MHz crystal clocks a six-bit synchronous counter decoded to produce forty states. The signal TOGGLE goes active each time counter state reaches 39. The TOGGLE signal causes the CLK_TX signal to change state every forty cycles of the C15_36MHZ clock. The period of CLK_TX is therefore eighty clock cycles or 5.2μs (65ns multiplied by 80). The CLK_RX clock signal is the inverse of CLK TX.

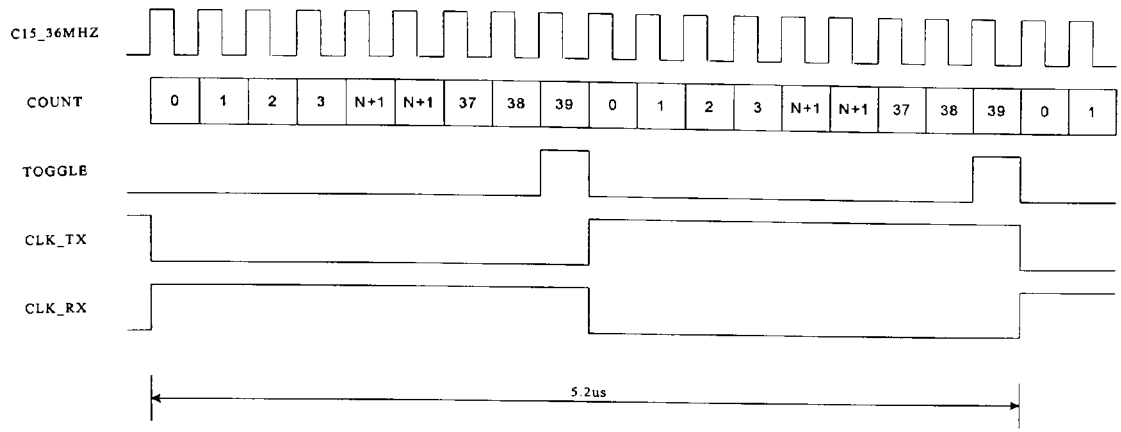


Figure 49 - The clock generation timing diagram.

The SBUS_BIT_CNT is used twice to generate the count sequences SBUS_CNT_NTTE and SBUS_CNT_TENT shown in figure 48a. The count sequence SBUS_CNT_NTTE is generated by U5 and is clocked by CLK_TX. The count sequence SBUS_CNT_TENT is generated by U7 and is clocked by CLK_RX. The TEST_DECODER block monitors the SBUS_CNT_NTTE sequence in order to derive a reset signal (RST_TENT_CNT) for U7. This ensures that there is a defined relationship between the two count sequences and that this is maintained throughout the circuit operation.

5.2 The SBITS_TX block.

The SBITS_TX block implements the S bits encoding for the maintenance channel in the NT to TE direction and provides signals to indicate the existence of sub-frame and multi-frame conditions. The block diagram for the SBITS_TX block is shown in figure 3 of appendix b and is composed of the SBITS_ENCODER block, the NTTECNT_DECODE block, and the FRAME_CNT block. Table 6 presents the

truth table defining the SC1 channel codes for each loopback situation. The SBITS_ENCODER block performs the logic encoding for the SC1 channel codes.

Loopback	SC14	SC13	SC12	SC11
LB1_I	1	0	1	1
LB2_I	1	1	0	1

Table 6 - The SC1 codes.

The FRAME_CNT implements a synchronous binary counter with a count sequence that ranges 0 to 20. The counter is advanced once per data frame by the signal START_NEXT_FRAME. The outputs signals, MULTIFRAME and SUBFRAME are decoded from the count sequence. MULTIFRAME will go active high for one frame every twenty frames when the count sequence is 0. SUBFRAME will go active every five frames when the count sequence is 0, 5, 10, and 15. The NTTECNT_DECODE block monitors the count sequence, SBUSCNT_NTTE and generates the signal START_NEXT_FRAME when the count sequence is forty-five.

5.3 The QBITS_RX block.

The QBITS_RX block reads the four Q bits from a multi-frame and determines the Q channel codes sent by the TE. The Q channel codes are decoded to establish the loopback request and this is broadcast to the other blocks in the design. The block diagram for the QBITS_RX block is shown in figure 4 of appendix b. The QBITS_RX block is composed of QBITS_RX_REG block, the

QBITS_RX_DECODER block, the QBITS_RX_LATCH block, the QBITS_RX_CNT block, the QBIT_DECODE, and the QBITS_RX_CON block. The QBITS_RX_REG is a serial in parallel out shift-register. The QBITS_RX_DECODER is a logic decoder that conforms to the truth table in table 7. The QBITS_RX_CNT is a synchronous binary counter with four states. When the fourth state is entered the counter signals that four Q bits have been extracted from the ISDN. The objective of the QBITS_RX block is to latch the Q bit when a frame with a Q bit is active, indicated by an active SUBFRAME signal. The QBIT_DECODE block monitors the SBUS_CNT_TENT count sequence and produces the signal QBIT when the count sequence is 13. Bit 13 is a Q bit when a sub-frame or multi-frame is active otherwise bit 13 is an Fa bit.

The QBITS_RX_CON implements the state-machine controller for the process. As each frame with a Q bit is identified, the controller shifts the next Q bit into the QBITS_RX_REG and advances the count sequence in the QBITS_RX_CNT block. When four Q bits have been loaded into the QBITS_RX_REG, the result of the QBITS_RX_DECODER is latched by the controller into the QBITS_RX_LATCH. The output signal LB1_R will be high if the TE requests a loopback on B1. The output signal LB2_R will be high if the TE requests a loopback on B2.

Loopback	QBIT4	QBIT 3	QBIT 2	QBIT 1
LB1_R	1	1	1	0
LB2_R	1	1	0	1

Table 7 - The Q bit codes.

5.4 The NT_LOOPBACK block.

The NT_LOOPBACK block is composed of the DMOD_DATA block, the B1_BUFFER block, the SBUSCNT_DECODE block and the NTLOOPBACK_CON block. . The block diagram for the NT_LOOPBACK block is in figure 5 of appendix b The SBUSCNT_DECODE block identifies the B channel data as shown in figure 48b and extracts the data from the line. The DMOD_DATA block converts the AMI data to binary data using the same technique as employed in the TE implementation. The B1_BUFFER blocks implement a serial in to parallel out shift-register. U2 stores the B1 channel data and U3 stores B2 channel data. The NTLOOPBACK_CON simply enables the U2 and U3 buffers during a loopback. In this way both B1 and B2 channels are extracted from the data frames received from the TE. The outputs from the U2 and U3 are sent to the INFOGEN_TENT block to be transmitted back to the TE in the next frame.

5.5 The INFOGEN_TENT block.

The INFOGEN_TETONT block in the NT generates the INFO4 signal toward the TE and performs the signal coding and timing violations in the same way as the TE. This block was already used in the TE design and some modifications are made to INFO3_GEN block to provide the functionality required for NT operation. The modifications effect the state-machine employed by the process. The supporting architecture remains the same as that used in the INFO3_GEN block for the TE case. The block diagram for the INFOGEN_TENT block is in figure 6 of appendix b. The INFO3_CON has been replaced with the INFO4_CON block. The

INFO4_CON block controls the coding process for the NT to TE direction of data transmission and the associated ASM chart is shown in figure 51. The same coding principle is employed by the INFO4_CON as that used by the INFO3_CON. The one-hot state encoding method was used during implementation. The following describes how each bit within the frame is coded.

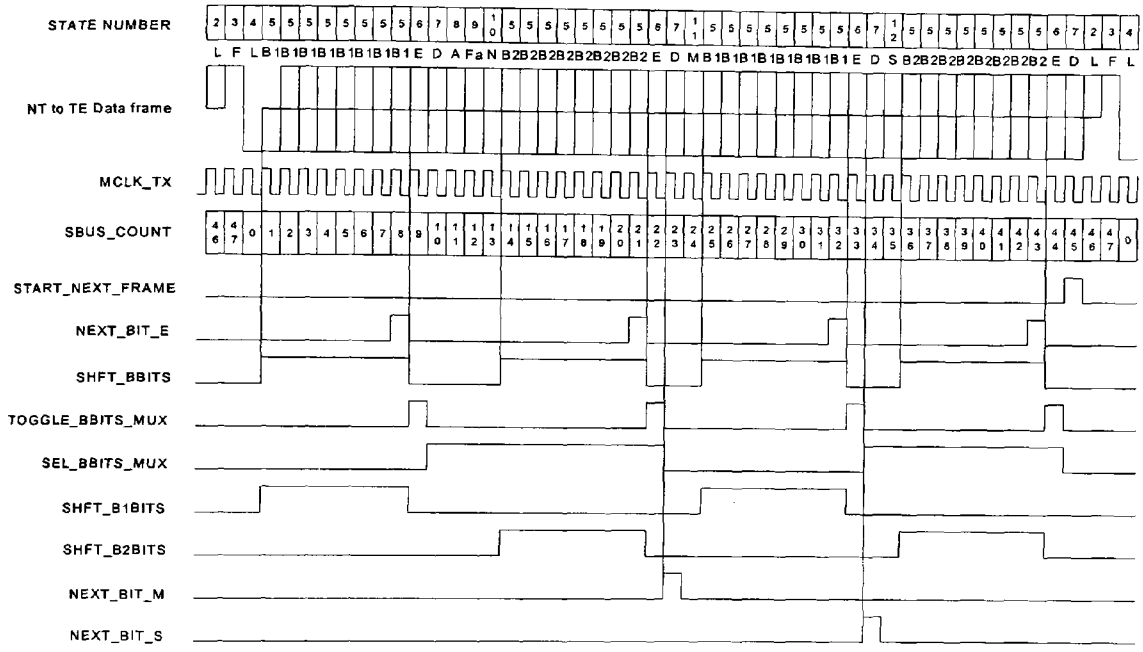


Figure 50 - The INFO4_GEN timing diagram.

Bit 45: This is coded in exactly the same way as the INFO3_CON block.

Bit 46: This is coded in exactly the same way as the INFO3_CON block.

Bit 47: This is coded in exactly the same way as the INFO3_CON block.

Bit 0: This is coded in exactly the same way as the INFO3_CON block.

Bit 1 to bit 8: This is coded in exactly the same way as the INFO3_CON block. In keeping with the ISDN naming conventions the signal NEXT_BIT_E is monitored to identify the end of the B channel.

Bit 9: This is coded in exactly the same way as the INFO3_CON block.

Bit 10: State 7 is entered while the count sequence decodes bit 10. State 7 processes the D channel bits. In this implementation the D channel is unused, however logic ones must be sent. The signal NEXT_D is forced to logic 1 in order to achieve this. The signal START_NEXT_FRAME is checked here and will be active if the end of the frame has been reached. The sequence returns to state 2 when this is the case. The signals NEXT_BIT_M and NEXT_BIT_S are decoded to indicate when the next bit should be coded as an M bit or an S bit. If neither of the cases already mentioned is identified then the Fa bit will be coded next.

Bit 11: During state 8 the A bit is coded as a logic 1. An INFO4 signal is identified by TEs when the A bit is coded as a logic 1.

Bit 12: State 9 exists when the TE to NT count sequence reads 12. State 9 processes the Fa bit. If a violation has not been made already it is forced during this state.

Bit 13: State 10 is entered and the N bit is processed. The N bit is coded so that it is always the binary opposite of the Fa bit. If a violation has not been made already it is forced during this state.

Bit 14 to bit 21: This is the first eight bits of the B2 channel and is coded in the same way as the INFO3_CON block.

Bit 22: State 6 is entered. This is coded in exactly the same way as the INFO3_CON block.

Bit 23: This is coded in exactly the same way as the INFO3_CON block.

Bit 24: State 11 is entered and the M bit is coded. If a multi-frame exists then this bit is coded as logic 1. Otherwise this is coded as logic 0.

Bit 25 to 32: This is the second eight bits of the B1 channel.

Bit 33: State 6 is entered and the E bit is coded.

Bit 34: State 7 processes the D channel bits.

Bit 35: State 12 is entered and an S bit is coded if the current frame is a sub-frame. A sub-frame is indicated when the signal SUBFRAME is active. Otherwise logic 0 is sent.

Bit 36 to 43: This is the second eight bits of the B2 channel.

Bit 44: State 6 is entered and the E bit is coded as logic 1.

Bit 45: This is coded in exactly the same way as the INFO3_CON block.

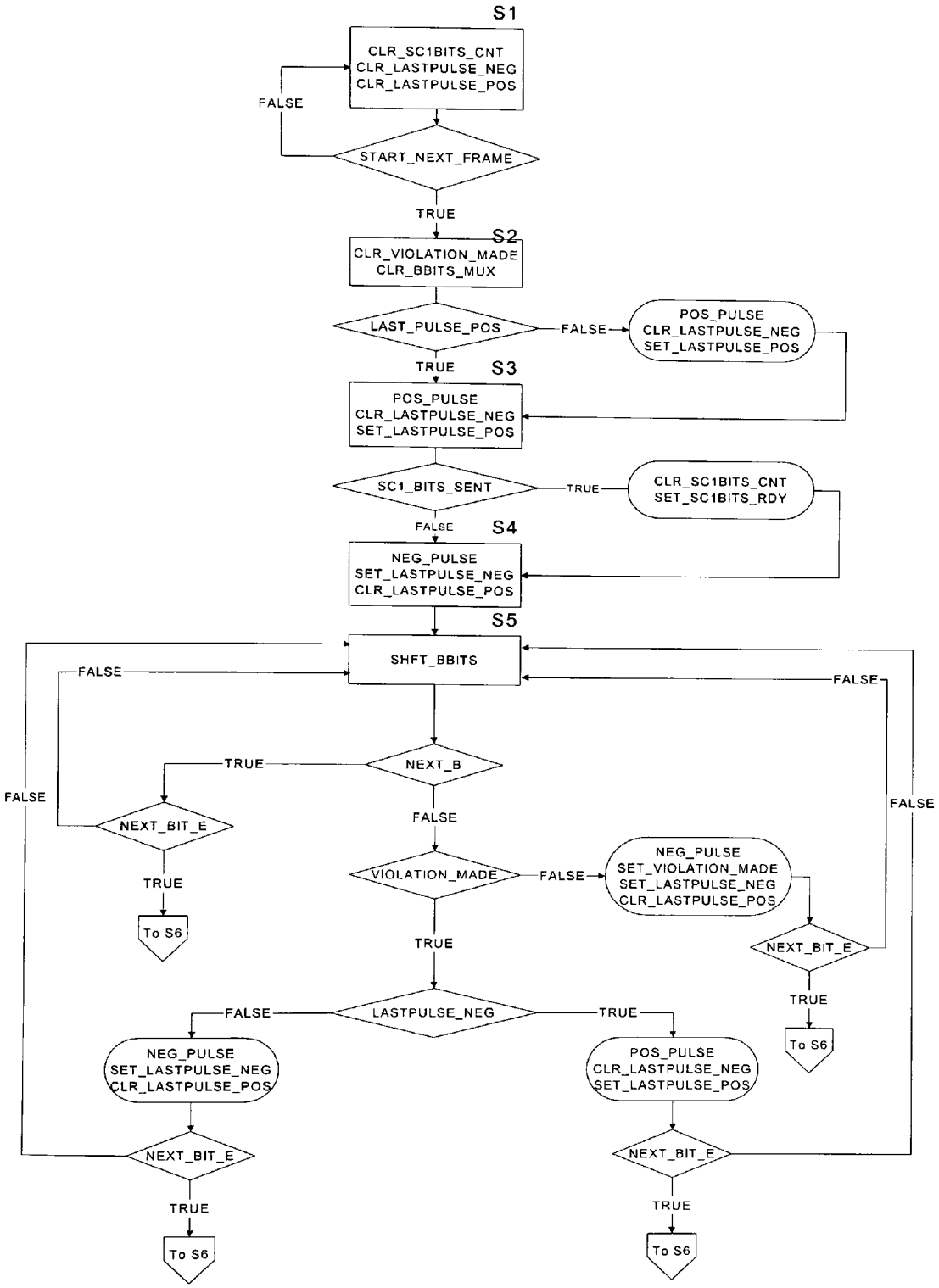


Figure 51a - The INFO4_CON block.

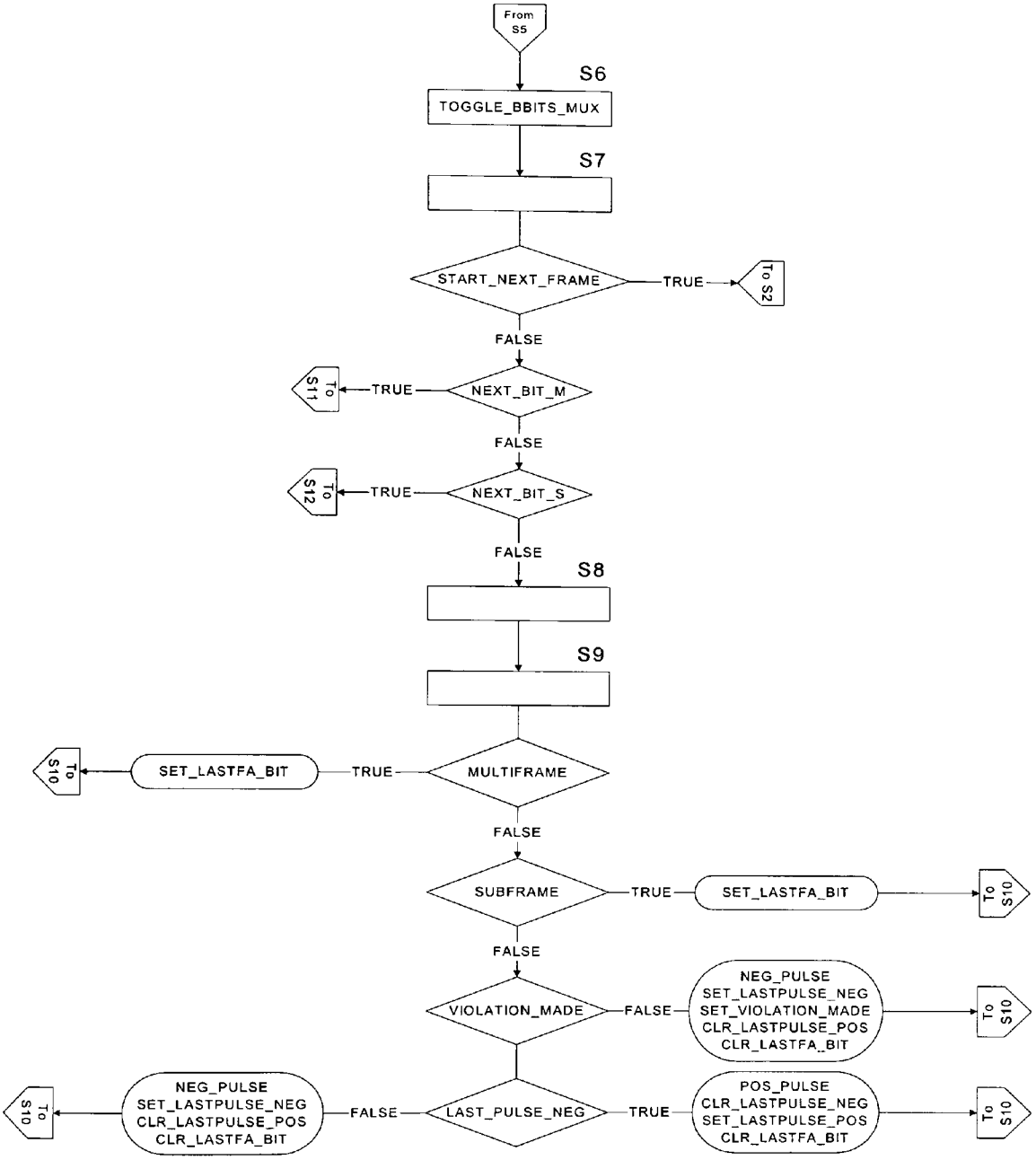


Figure 51b - The INFO4_CON block.

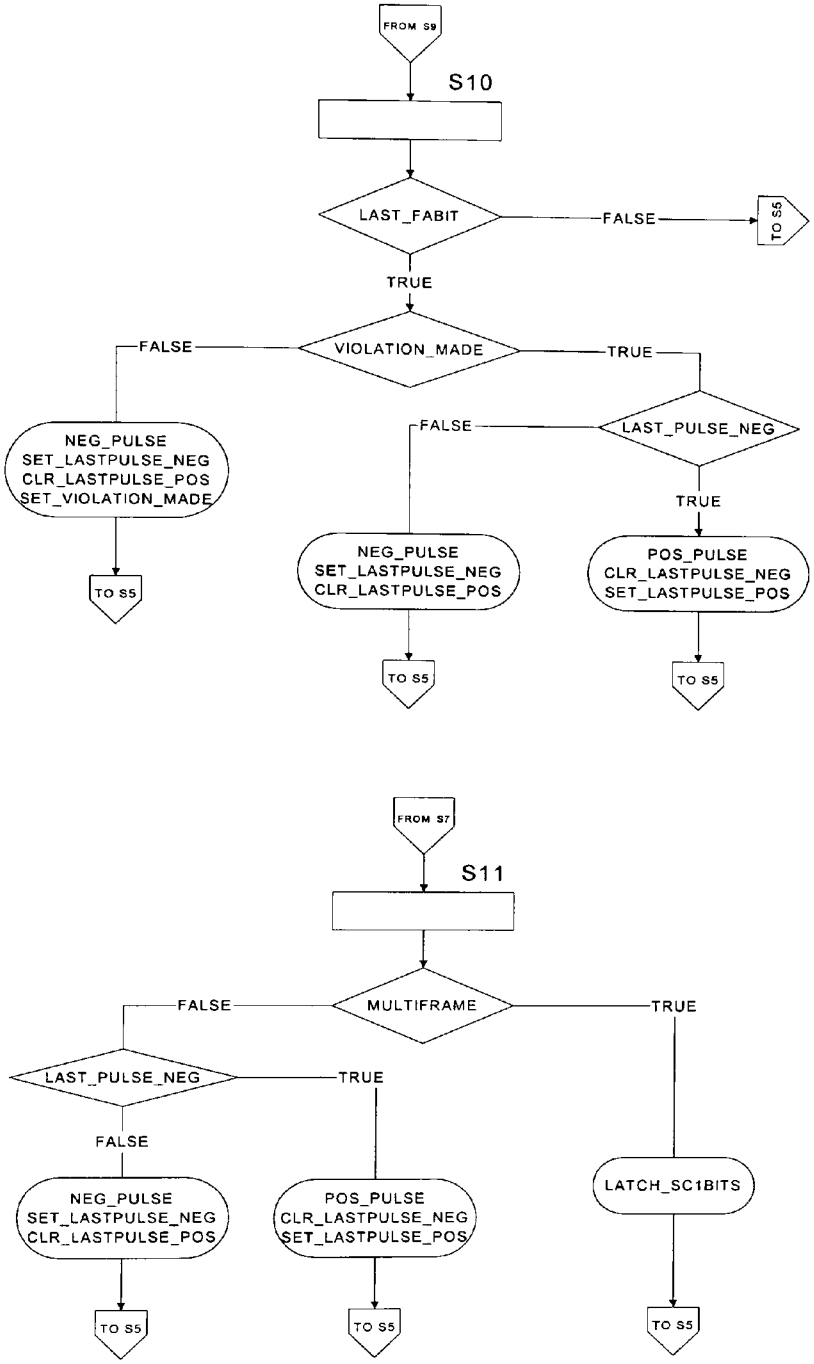


Figure 51c - The INFO4_CON block.

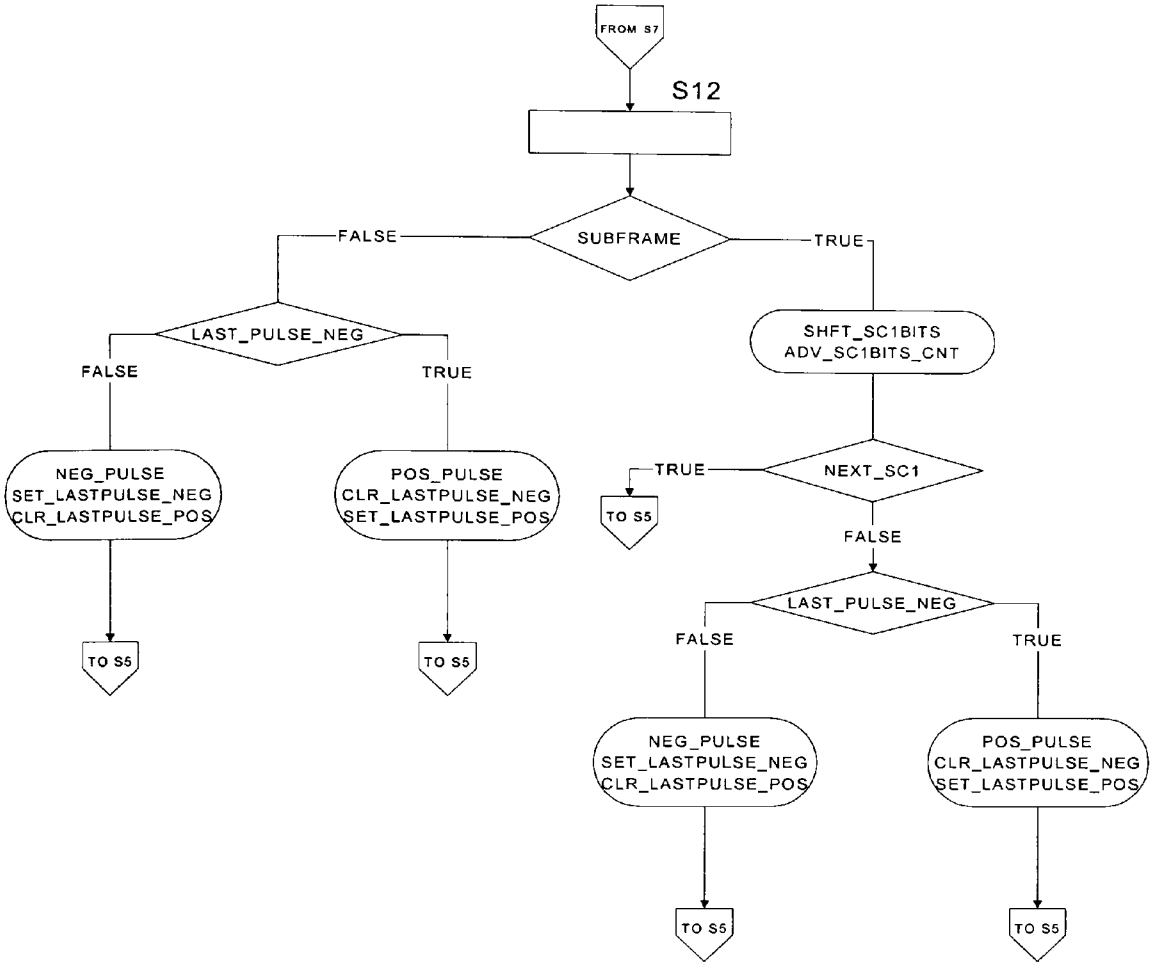


Figure 51d - The INFO4_CON block.

6 The design verification and results.

The results presented in this section provide the proof relating to maintenance channel synchronisation and loopback functionality of the Terminal equipment. The design and implementation effort in each case has involved an iterative cycle of testing and design review exercises for each block in the design. The main objective of the test procedure is to validate the Terminal equipment operation. It is valid to state that the Network terminator design employed to facilitate the test procedure is also validated during the TE test procedure. This is due to the relationship that the basic-rate ISDN TE and NT share. The TE and NT design efforts have been executed separately and strictly in accordance with ITU-T specifications. The final system test and validation focuses on the ability of the TE to synchronise to the data frames from the NT, synchronise to the maintenance channel embedded in the NT data frames and effect a successful loopback. Once a loopback is in place the bit error rate monitoring capability of the system is verified.

A printed circuit board (PCB) was designed specially for this project and offers all the facilities to support layer-1 basic-rate ISDN operation. There are two FPGAs on the PCB. One of the FPGAs is configured to operate in TE mode while the other is configured as an NT. The NT configuration is connected to the TE configuration via transformers fitted on the PCB. The results from the test and validation exercise for the TE data frame synchronisation and phase locked loop operation have already been presented. The relationship between the NT to TE and TE to NT directions of data transmission is presented in the oscilloscope plot in figure 54. The NT to TE direction of data transmission is shown on channel 1. The TE to NT direction is plotted on channel 2. It is expected that a two-bit offset should

exist between both directions of transmission. The comparison is made between the F bits of each channel. The F bits can be identified once the two sets of coding violations are located. The broken curser is aligned with the F bit on channel 1 and the solid curser is aligned with the F bit on channel 2. The offset measured is two bit periods as shown. The verification process involved detailed scrutiny of the data frame structures for both directions of transmission and it was concluded that the TE synchronises to the NT and transmits data frames towards the NT in accordance with that specified in I.430.

The test set-up for the maintenance channel synchronisation and loopback verification is shown in the top-level block diagram for the TE Firmware design. Test control signals are connected to pins on the FPGA. The input pin number 23 is connected to an external switch and the switch state controls the active loopback. If pin 23 is low, a request for a loopback in B1 is sent to the NT. If pin 23 is high, a request for a loopback in B2 is sent to the NT. When synchronisation is established with the NT, the SYNC signal from the SBUS_DLL block is broadcast to the other logic blocks in the design. When SYNC is high the INFOGEN_TENT block ceases transmission of INFO1 and INFO3 generation is enabled. The LOOPBACK block attempts to synchronise with the maintenance channel from the NT when SYNC is high.

During the verification process the signals LB1_I and LB2_I were monitored. An active high on LB1_I indicates that the NT has placed a loopback on the B1 channel. An active high on LB2_I indicates that the NT has placed a loopback on the B2 channel. A logic analyzer was used to check that the correct S channel and Q channel codes were sent and received during the maintenance channel synchronisation process. The state of the LB1_I and LB2_I signals were

captured during the test procedure with an oscilloscope. In figure 55, channel 1 represents the signal LB1_I and channel 2 is LB2_I. The signal LB1_I is active indicating that a loopback was successfully placed on B1. Figure 56 shows the active LB1_I signal along with verification of PRBS in the B1 channel. Using the same verification procedure, figures 57 and 58 indicate that a loopback was successfully placed on B2.

Once the maintenance channel synchronisation was verified the scrutiny was focused on the PRBS transmitter and receiver circuits. PRBS transmission will begin when the PRBS circuitry is reset. Pin number 25 is connected to an external switch to facilitate this. Firstly a loopback was placed on B2 and PRBS transmission was enabled. Figure 59a shows a data frame in the TE to NT direction of data transmission. Channel 1 shows the transmit clock and channel 2 shows the TE to NT data frame. It is impossible to represent the random nature of the data here, however the procedure adopted during the validation procedure will be discussed and a conclusion made.

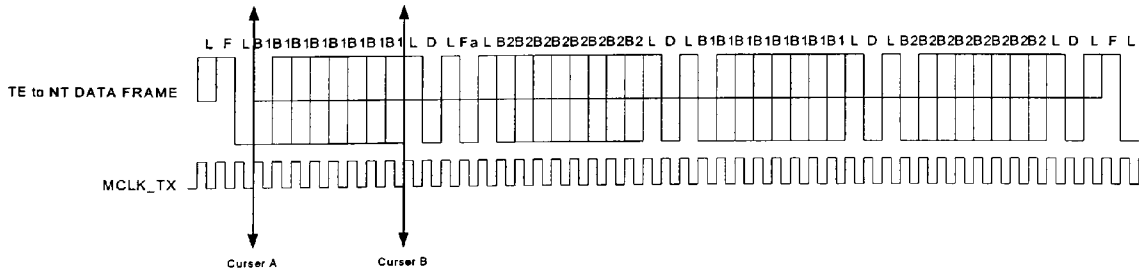


Figure 53 - The TE to NT data frame with transmit clock.

Figure 53 shows the data frame generated by the TE toward the NT with the transmit clock. The verification procedure involves monitoring the data frames and assessing if the TE is inserting the PRBS sequence in the correct channel. The

transmit clock is required for oscilloscope triggering purposes and to enable precise location of the B channels within the frame. The oscilloscope was used to monitor the signals and the oscilloscope cursers were used in the measurement procedure. For example, to locate the B channel, the FL pair is firstly located. The first eight bits of the B1 channel will exist over a time span of 41.6μs (5.2μs X 8). Step one involves positioning curser A after the FL pair, as shown in figure 53. Curser B is then positioned such that the distance between the two equals 41.6μs. The cursers will now window the first eight bits of the B1 channel. The existence of random data will be clearly seen. The cursers may then be moved along the frame using the edges of the transmit clock to identify individual bits within the data frame. Figure 59a shows the verification procedure for PRBS in the first eight bits of the B1 channel. Curser A may then be positioned at the end of the B1 channel and curser B may be moved a distance of sixteen bits or 83μs (5.2μs X 16) to locate the beginning of the next eight bits of the B1 channel. This is shown in figure 59b. In this way the ability of The TE to correctly insert PRBS in the correct locations within the data frame is verified. Figures 60a, 60b, 60c and 60d show the verification procedure for the B2 channel.

The current test set up should provide a perfect transmission line between the NT and TE. It is not expected that the PRBS receiver should indicate bit errors during a bit error rate test. To verify this, the PRBS receiver was tested to ensure that synchronisation was maintained consistently over a long time period. Causing bit errors and monitoring the PRBS error counter with a logic analyzer verified the integrity of the loopback. This was achieved with the ERR_INJ block. The ERR_INJ block was designed to facilitate the test and validation process. An external frequency generator was used to generate a 1Hz clock signal. The

ERR_INJ block was employed to monitor the positive going edge of a 1Hz clock signal and inject a bit error into the transmitted PRBS sequence. The PRBS receiver identified the bit error in the received sequence and this was registered by advancing the PRBS bit error counter. A logic analyzer was connected to the PRBS error counter output to verify that this was the case. The PRBS receivers bit error signal was monitored over sixteen hours and figure 61 shows the PRBS receivers bit error signal on channel two. This test was performed to verify the integrity of the loopback and it was concluded that the Firmware for TE worked as expected.

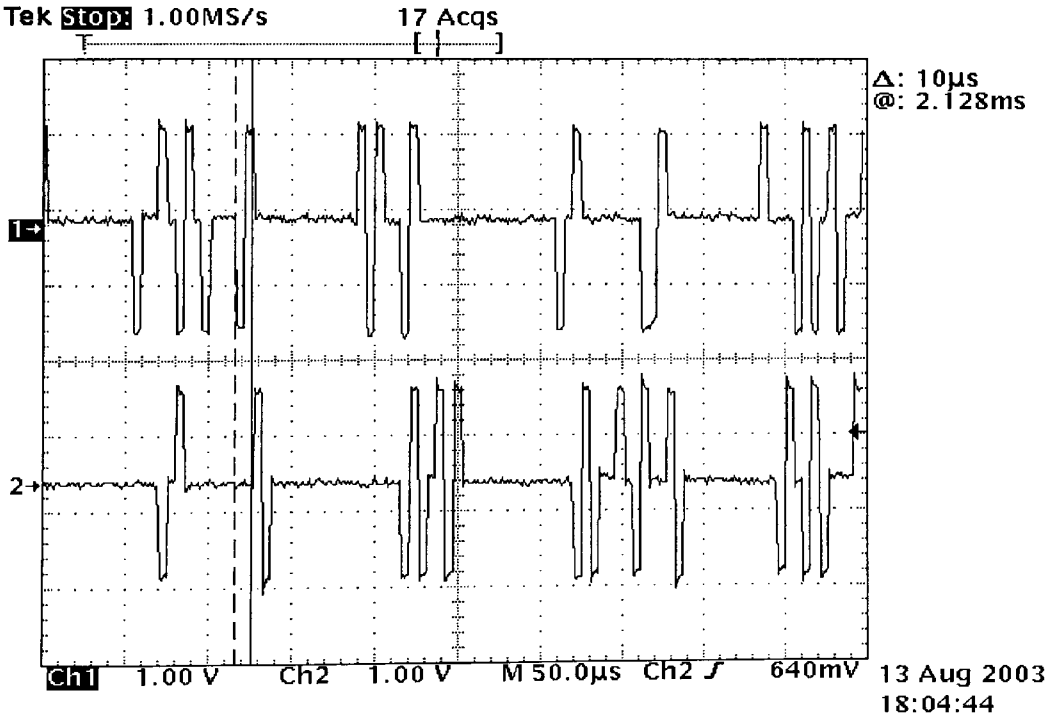


Figure 54- The NT to TE and TE to NT data frame.

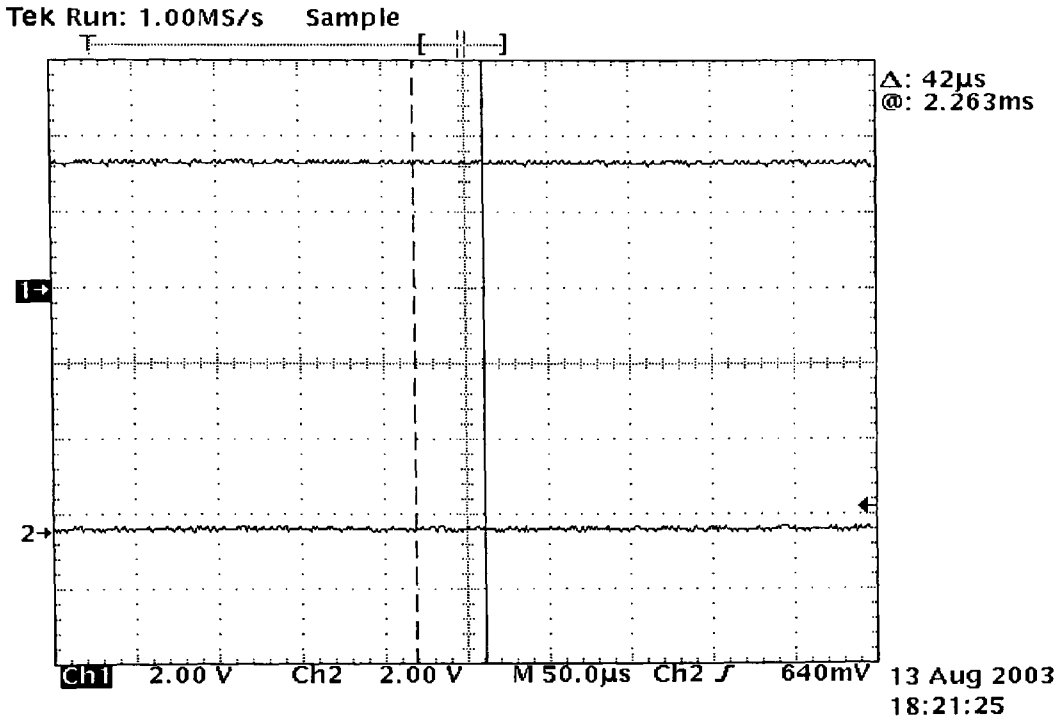


Figure 55 - The LB1_I and LB2_I signal states.

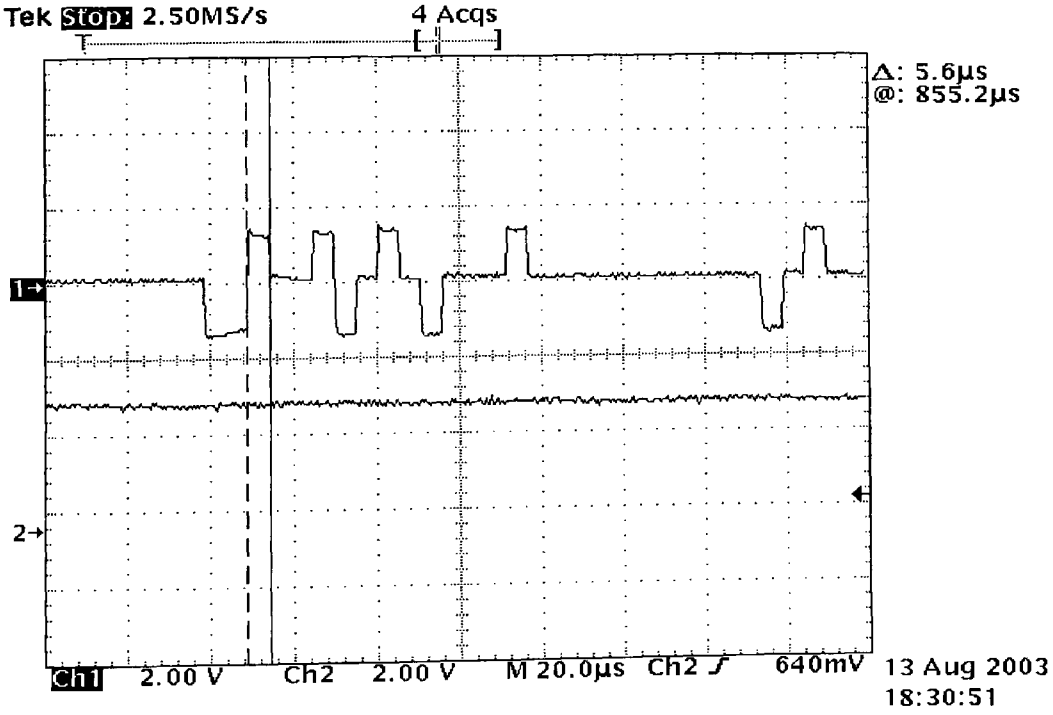


Figure 56 - B1 channel loopback indication .

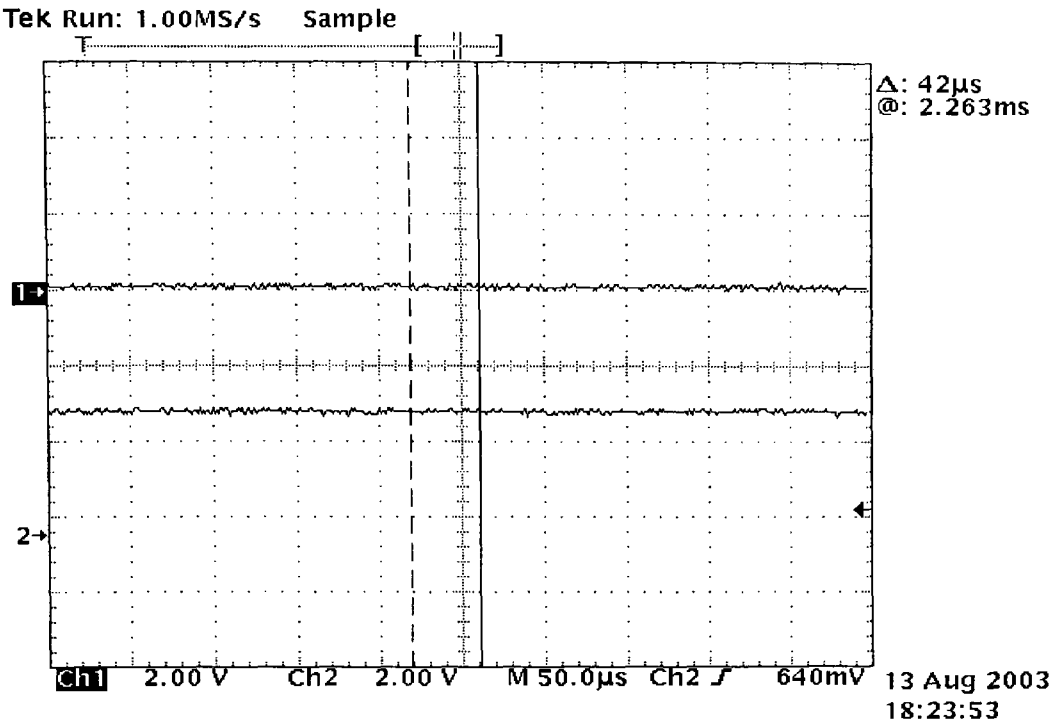


Figure 57 - The LB1_I and LB2_I signal states.

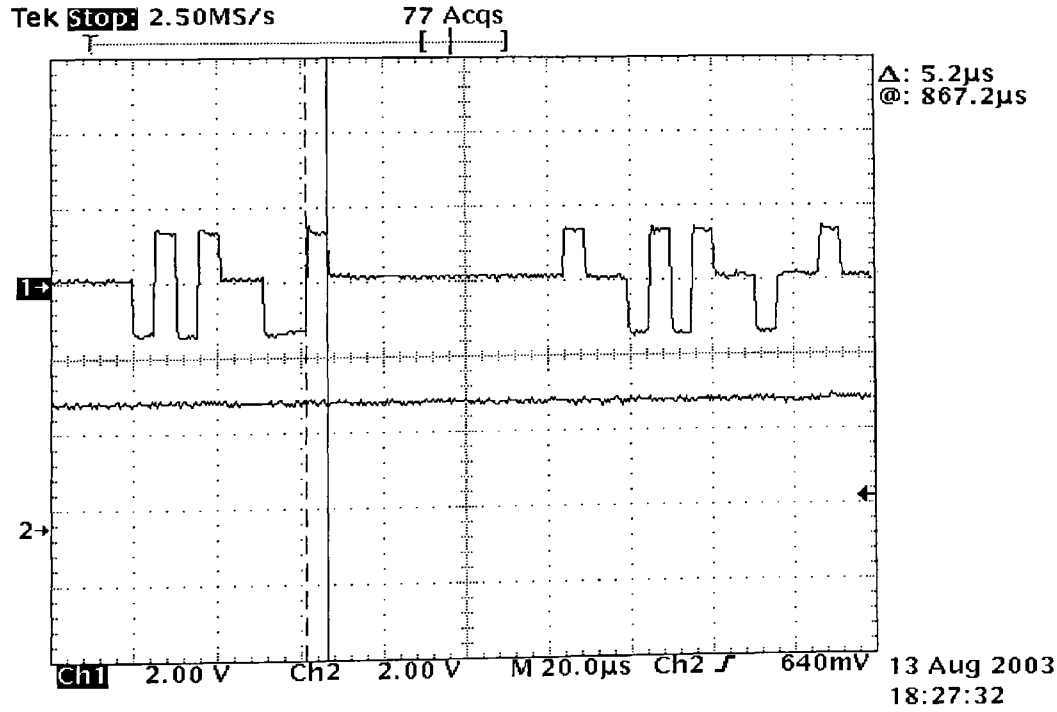


Figure 58 - B2 channel loopback indication .

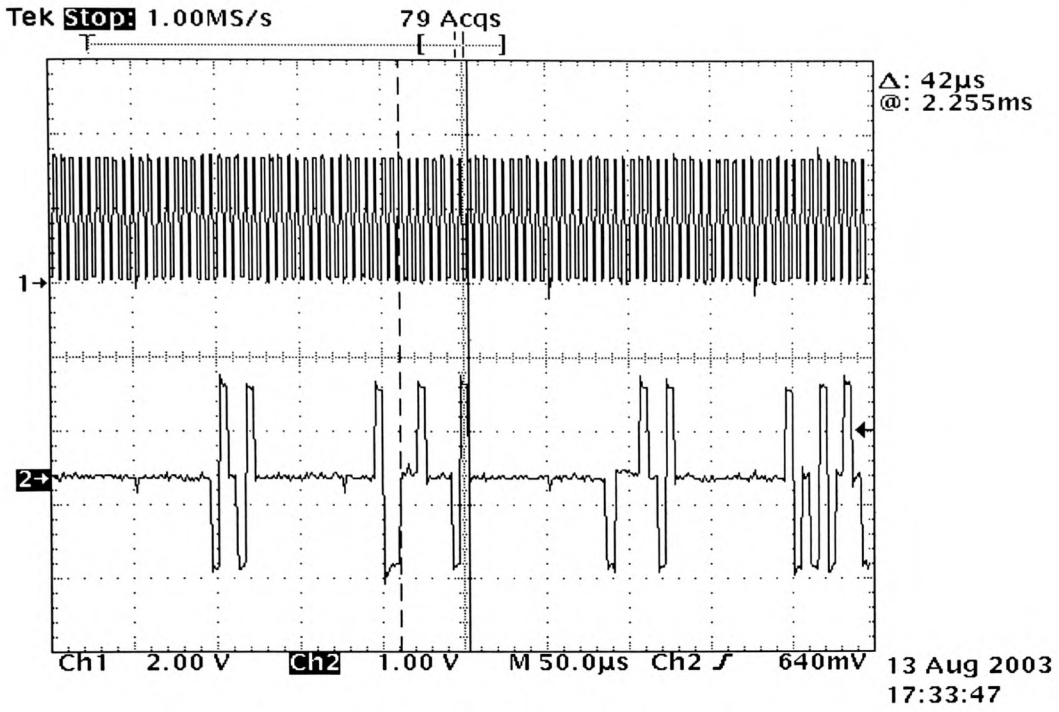


Figure 59a - PRBS in the B1 channel.

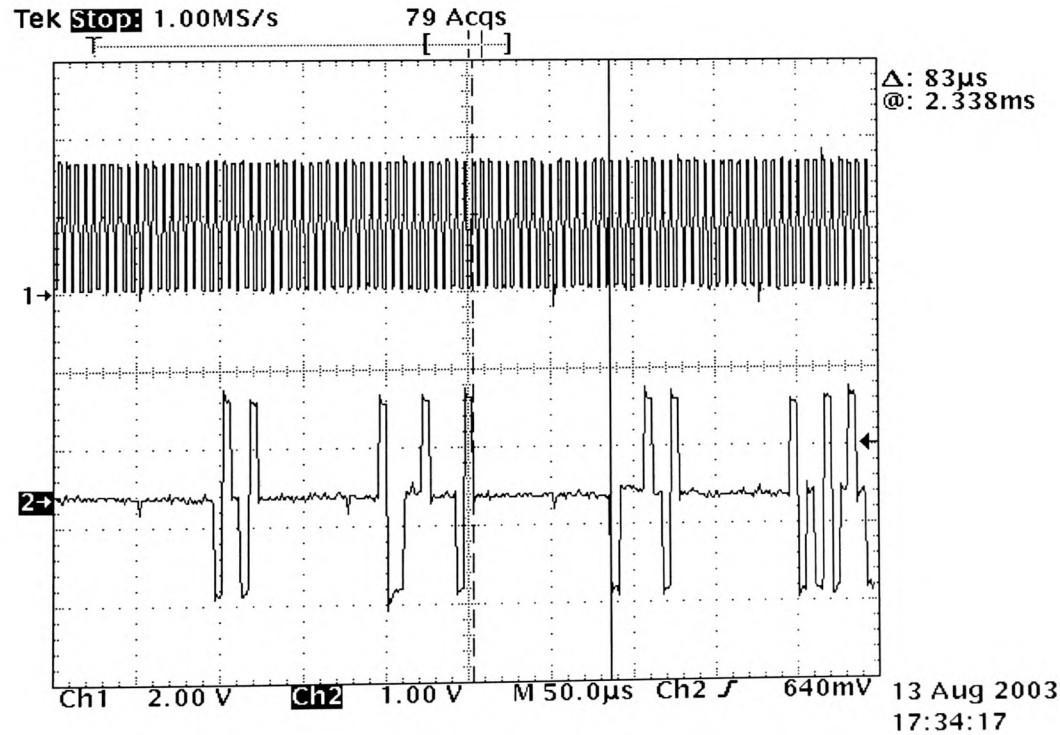


Figure 59b - PRBS in the B1 channel.

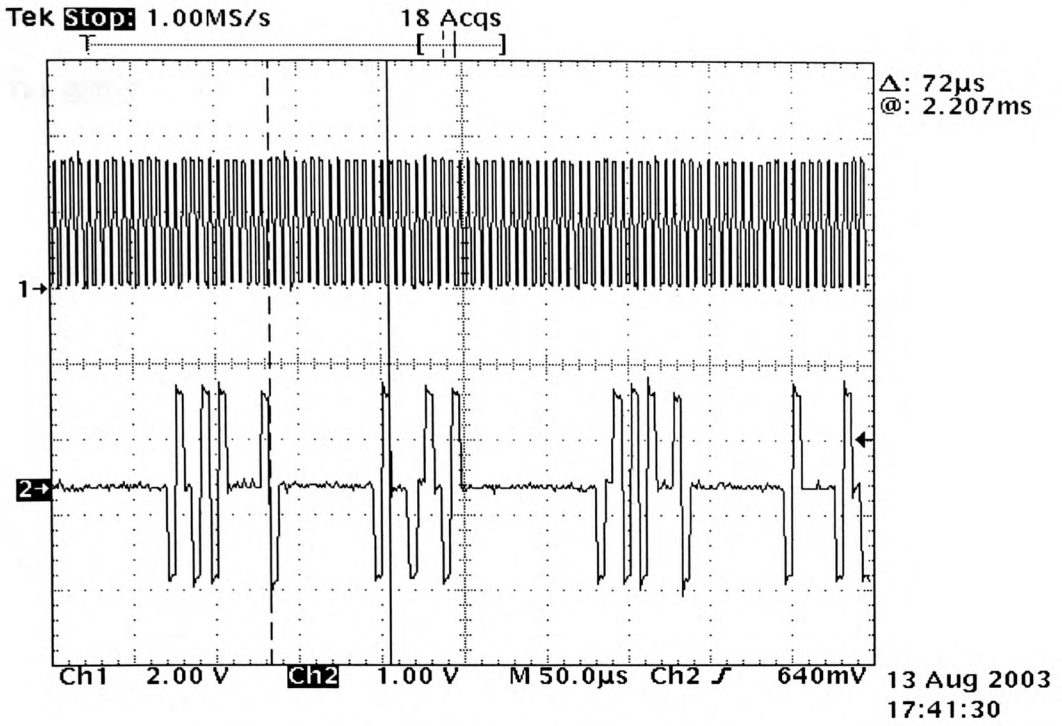


Figure 60a - PRBS in the B2 channel.

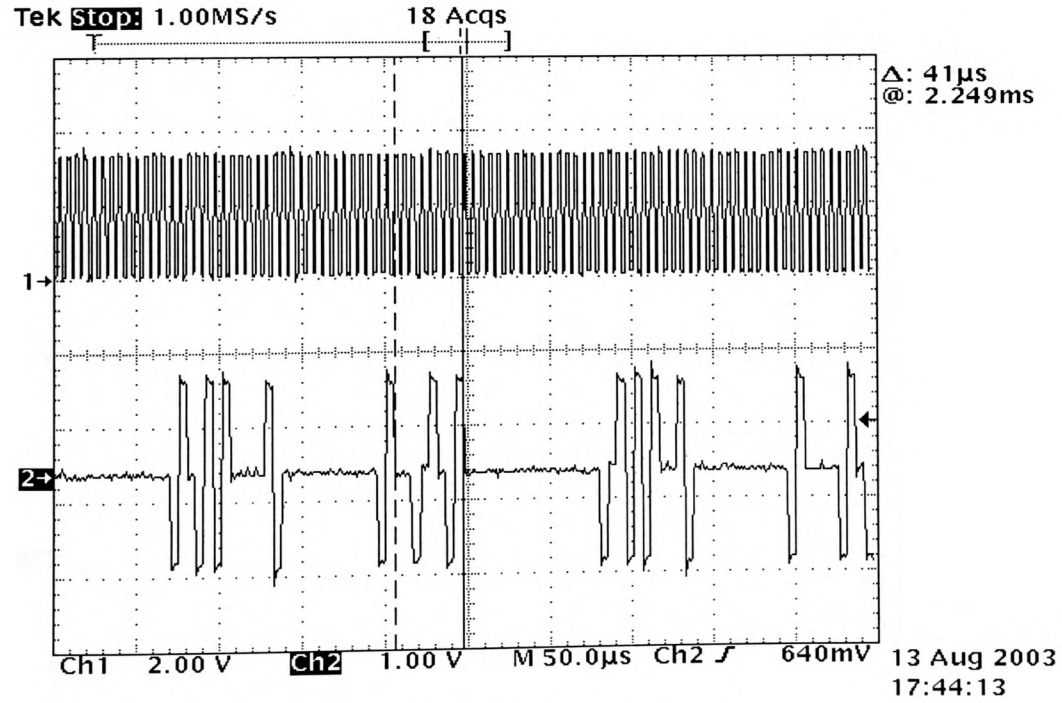


Figure 60b - PRBS in the B2 channel.

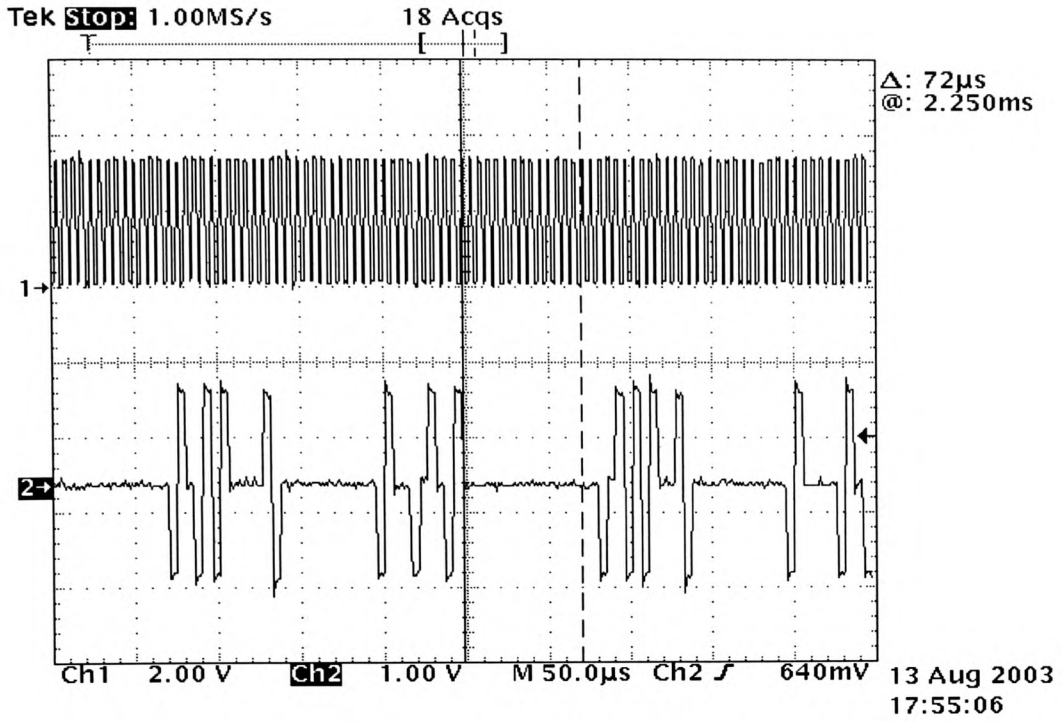


Figure 60c - PRBS in the B2 channel.

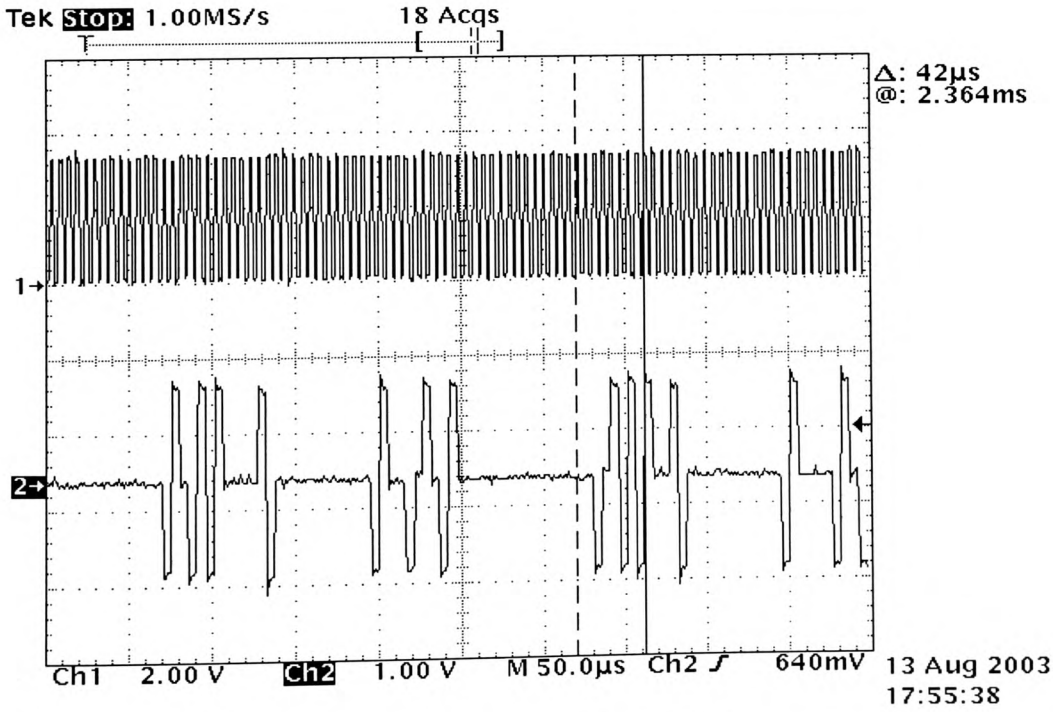


Figure 60d - PRBS in the B2 channel.

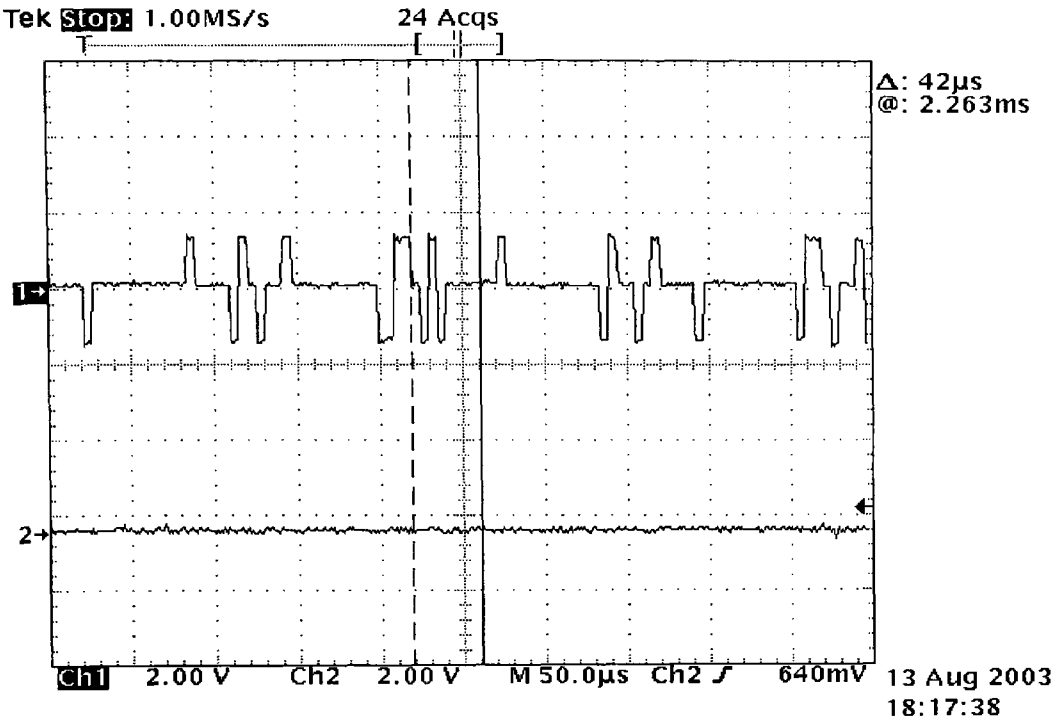


Figure 61 - PRBS receiver error monitoring.

6.1 Implementation details.

The implementation details present the FPGA device utilisation summary for both the NT and TE implementations. The NT firmware was targeted at a Xilinx 4005XL FPGA while the TE was implemented on a Xilinx 4010XL. The configurable logic blocks (CLB) utilisation represents the best summary for device utilisation and this is presented for both cases in the following.

Device utilization summary for the NT Firmware.

Number of External IOBs 24 out of 61 39%
Flops: 0
Latches: 0
Number of Global Buffer IOBs 1 out of 8 12%
Flops: 0
Latches: 0
Number of CLBs: 163 out of 196 83%

Total Latches: 0 out of 392 0%
Total CLB Flops: 131 out of 392 33%
4 input LUTs: 297 out of 392 75%
3 input LUTs: 42 out of 196 21%
Number of BUFGLSs 3 out of 8 37%
Number of STARTUPs 1 out of 1 100%

Device utilization summary for the TE Firmware.

Number of External IOBs 22 out of 61 36%
Flops: 0
Latches: 0
Number of Global Buffer IOBs 1 out of 8 12%
Flops: 0
Latches: 0
Number of CLBs 262 out of 400 65%
Total Latches: 0 out of 800 0%
Total CLB Flops: 222 out of 800 27%
4 input LUTs: 480 out of 800 60%
3 input LUTs: 47 out of 400 11%
Number of BUFGLSs 3 out of 8 37%
Number of STARTUPs 1 out of 1 100%

7 Conclusions and further work.

The overall objective of this project was to integrate the layer one circuits required to establish a loopback at the network termination point, with the PRBS circuits required to test the data transmission integrity of the network. The work concentrated on the design of both TE and NT equipment. The TE design exercise succeeded in the development and implementation of both the analogue and digital circuits to realise a BERT function. The NT design exercise focused on the development of the features required for TE design verification. During the design exercise both the analogue and digital circuitry for the NT was developed. The TE and NT designs were populated on a single PCB for testing. ISDN Transformers were included in the design of both solutions to allow galvanic isolation when data transmission between the NT and TE took place. This presented a realistic situation for testing and validation purposes. The FPGA was utilised as the core component in both designs. The following conclusions can be drawn from the work.

7.1 The conclusions.

1 The main objective was to provide a solution that achieves greater circuit integration than currently deployed ISDN bit error rate testers. The integration of the S Bus Interface circuitry, maintenance channel control circuitry, and the PRBS circuitry was achieved and thus a fully integrated S bus transceiver suitable for BER testing was produced. The analogue electronics required for the ISDN transceiver was also developed and produced.

2 The objective was to challenge the design philosophy currently adopted by test equipment designers by presenting an alternative design approach. ISDN testers may employ an ISDN S bus transceiver with maintenance channel functionality and either an FPGA or a BERT chip to realise the BERT feature. The ability to remove the S Bus transceiver and BERT chip from a testers bill of materials and replace these components with a single FPGA reduces cost. When this same approach is applied to other ISDN test features it is possible to achieve a further reduction in cost. The adoption of the FPGA as the core component for this project demonstrated that increased feature integration with the associated cost reductions could be achieved. It was also demonstrated that the analogue electronic design required for ISDN transceivers could be achieved with active electronics. During the design the ability to reconfigure the FPGA was exploited to allow rapid development time. This increased design flexibility demonstrated that the development cost could be kept to a minimum and shows that a considerable reduction in cost can be achieved over an ASIC design approach. This rapid development time has shown that the fast time to market for ISDN test equipment can be achieved. The ability to reconfigure the design when the test equipment is in the field extends the product life cycle.

3 An NT supporting the maintenance channel and providing loopback facilities was required to verify the TE design. Both the analogue and digital electronics for the NT was successfully developed. It was demonstrated that the NT design successfully transmitted the ISDN S bus signals with the required timing information to allow verification of the TE synchronisation and ADPLL circuits. The design and implementation of the circuitry to provide maintenance channel facilities was successful. The NT was designed to generate an S channel in the NT

to TE direction of data transmission and to interrogate and respond to the Q channel in the TE to NT direction. The NT successfully provided a loopback facility for both the B1 and B2 channels and was instrumental in the testing and validation exercise.

4 The objective to design an integrated synchronisation circuit with a digital ADPLL was successfully achieved. The clock extraction jitter was measured against the jitter requirements as defined by recommendation I.430 and it was shown that compliance with the jitter requirements was achieved. The maintenance channel control circuitry for the TE was successfully designed and implemented. The specifications were drawn from recommendation I.430 and it was verified during bench testing exercises that the circuit consistently established a loopback in the NT when a request was made by the TE.

5 The PRBS transmitter and receiver circuits were successfully designed and integrated with the layer one circuits as already outlined above. It was demonstrated that the PRBS circuitry could be integrated with layer one transceiver functions thus achieving the main objective.

7.2 Future work.

1 The current design provides the analogue and digital hardware required to achieve a fully integrated S Bus ISDN transceiver. The design has not been extended to encompass the development of the D channel controller circuitry. ISDN telephone calls are established via the D channel. The D channel controller circuits facilitate call setup on the ISDN. This feature is required to dial and connect to

another TE either on the same network or on another network. A voice or data call can then be made.

2 Once a telephone call is active data will flow on the B channels. In order to facilitate this operation elastic buffering facilities are required by the TE. Data to be transmitted and received are held in these buffers until processing is complete. The design effort did not focus on this area. If a voice or data call test feature is required then elastic-buffering circuits will be required.

3 The Layer one state machine was not required by this application. The development of an I.430 compliant ISDN transceiver for ISDN signal transmission applications will require one. The integration capabilities of the FPGA currently used for the BERT application may not be sufficient to include this feature. The footprints are identical for all the FPGAs in the Xilinx XL/Spartan family. Therefore, upgrading to an FPGA with more facilities is achievable.

References.

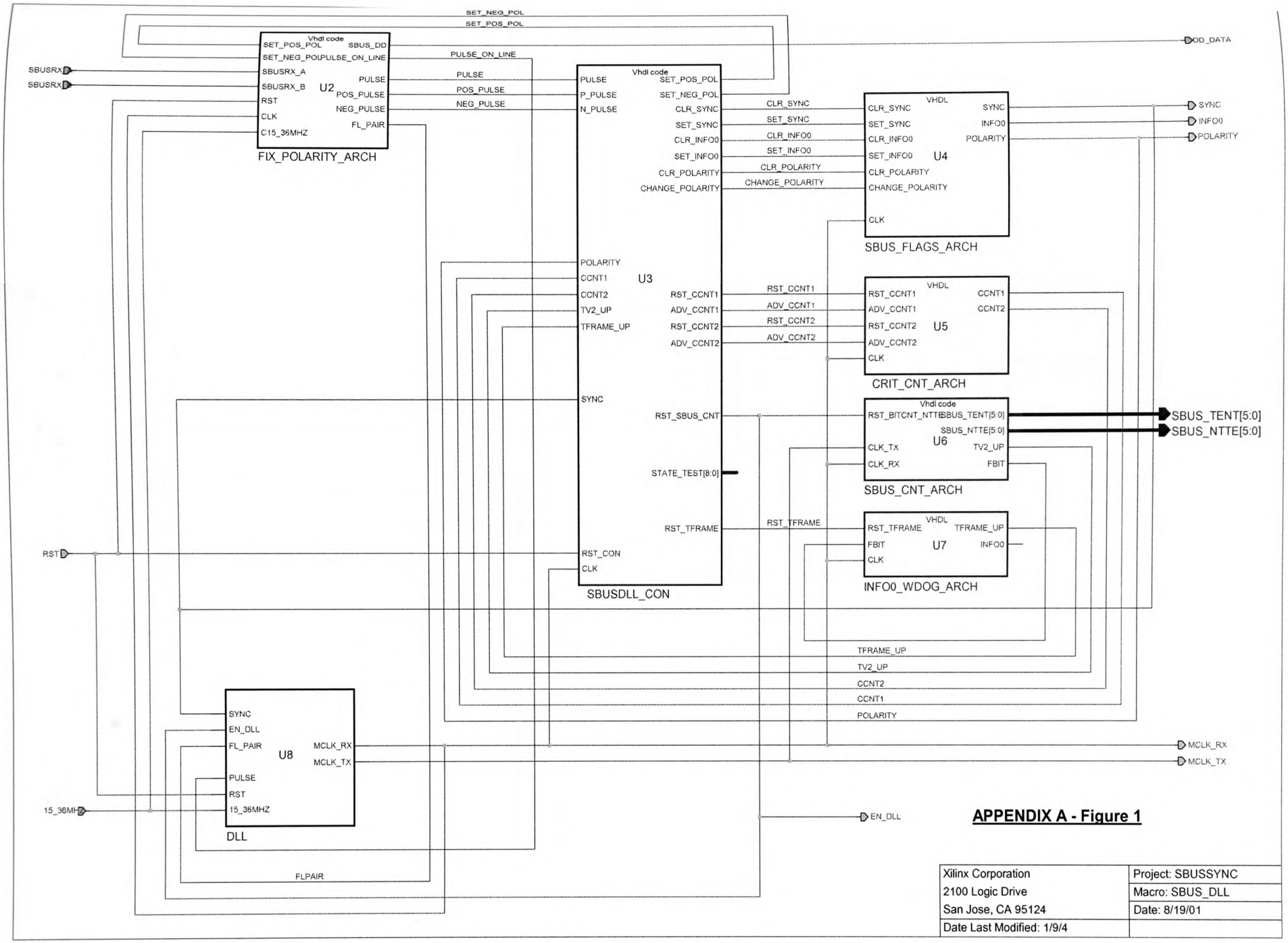
[1] Siemens, The IOM-2 Interface Reference Guide, Version 01.90, March 1991.

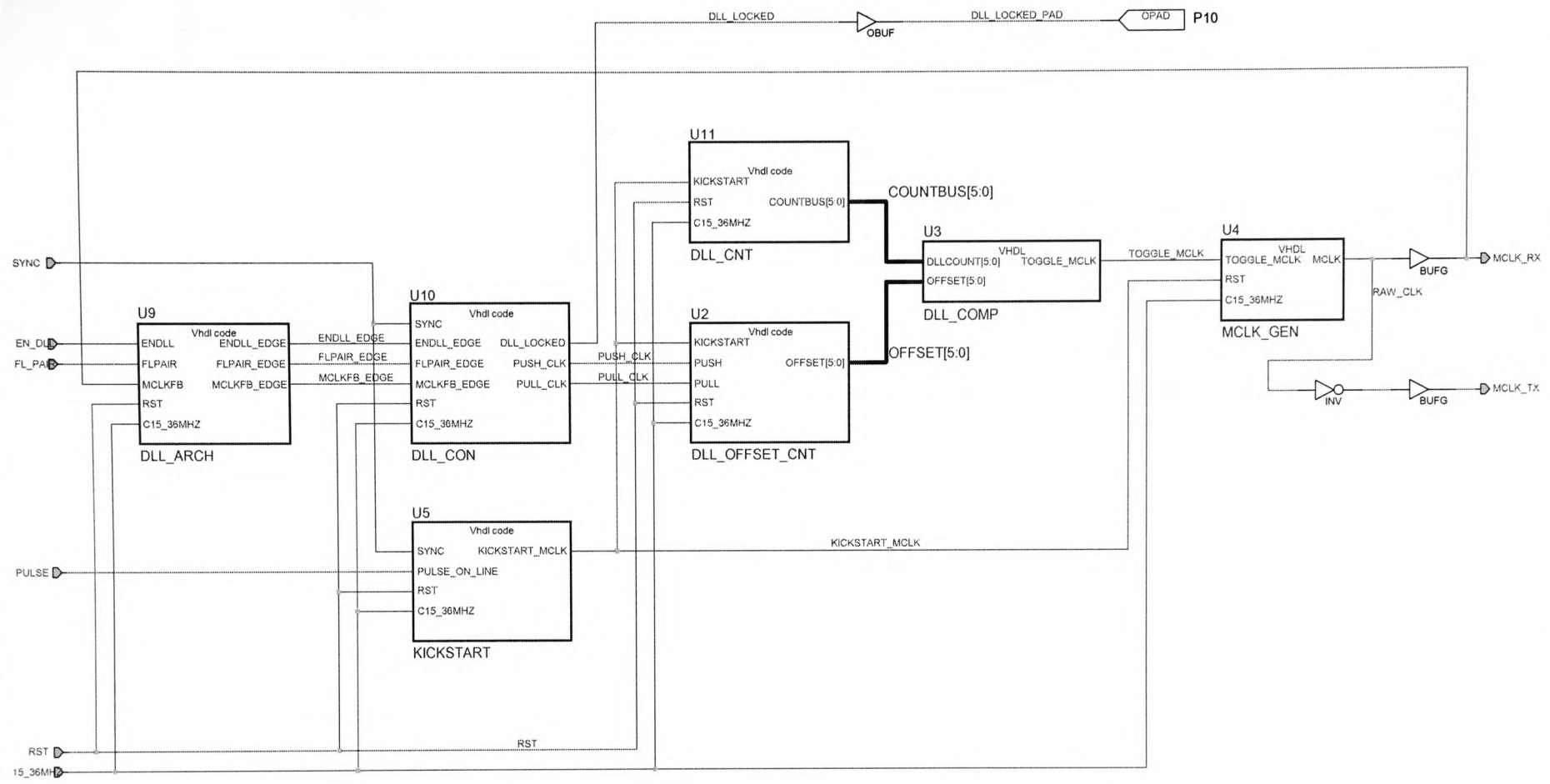
[2] International Telecommunications Union (ITU-T), Recommendation I.430, Basic User-Network Interface – Layer 1 Specification, 1993.

[3] Fairchild Simiconductor, BC546/547/548/549/550 data sheet, Rev A2, August 2002.

Appendix A.

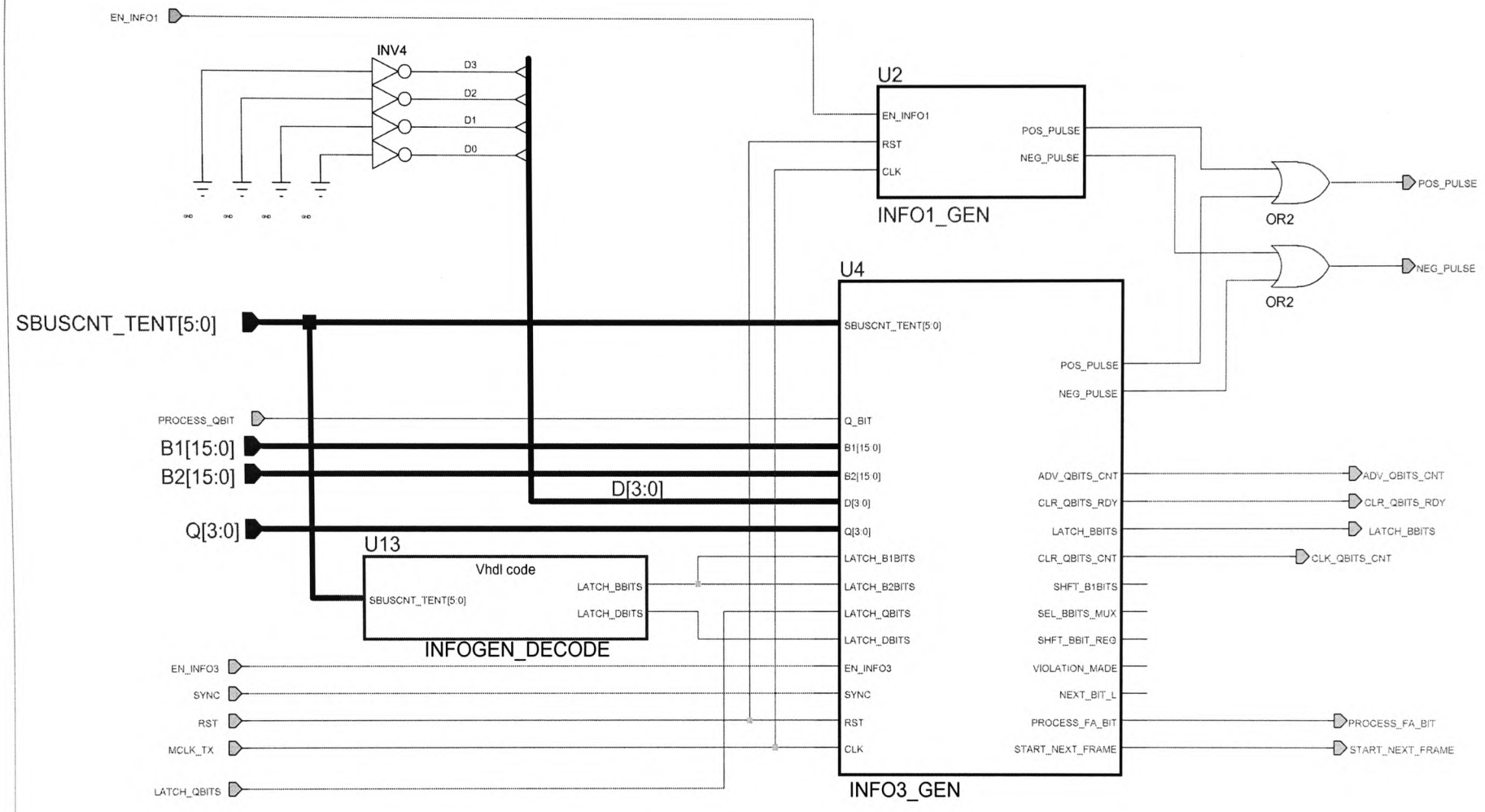
The FPGA firmware block diagram for the TE.


SET_NEG_POL
SET_POS_POL
OD_DATA
Vhdl code
SET_POS_POL
SET_NEG_POL
SBUS_DD
PULSE_ON_LINE
SBUSRX
SBUSRX
SBUSRX_A
SBUSRX_B
RST
CLK
C15_36MHZ
U2
PULSE
POS_PULSE
NEG_PULSE
FL_PAIR
FIX_POLARITY_ARCH
PULSE
POS_PULSE
NEG_PULSE
Vhdl code
PULSE
P_PULSE
N_PULSE
SET_POS_POL
SET_NEG_POL
CLR_SYNC
SET_SYNC
CLR_INFO0
SET_INFO0
CLR_POLARITY
CHANGE_POLARITY
CLR_SYNC
SET_SYNC
CLR_INFO0
SET_INFO0
CLR_POLARITY
CHANGE_POLARITY
VHDL
CLR_SYNC
SET_SYNC
CLR_INFO0
SET_INFO0
CLR_POLARITY
CHANGE_POLARITY
CLK
SYNC
INFO0
POLARITY
U4
SBUS_FLAGS_ARCH
SYNC
INFO0
POLARITY
POLARITY
CCNT1
CCNT2
TV2_UP
TFRAME_UP
U3
RST_CCNT1
ADV_CCNT1
RST_CCNT2
ADV_CCNT2
RST_CCNT1
ADV_CCNT1
RST_CCNT2
ADV_CCNT2
VHDL
RST_CCNT1
ADV_CCNT1
RST_CCNT2
ADV_CCNT2
CLK
CCNT1
CCNT2
U5
CRIT_CNT_ARCH
SYNC
RST_SBUS_CNT
STATE_TEST[8:0]
Vhdl code
RST_BITCNT_NTTE
CLK_TX
CLK_RX
SBUS_TENT[5:0]
SBUS_NTTE[5:0]
U6
TV2_UP
FBIT
SBUS_TENT[5:0]
SBUS_NTTE[5:0]
SBUS_CNT_ARCH
RST_TFRAME
RST_TFRAME
VHDL
RST_TFRAME
FBIT
CLK
TFRAME_UP
INFO0
U7
INFO0_WDOG_ARCH
RST
RST_CON
CLK
SBUSDLL_CON
TFRAME_UP
TV2_UP
CCNT2
CCNT1
POLARITY
SYNC
EN_DLL
FL_PAIR
PULSE
RST
15_36MHZ
U8
MCLK_RX
MCLK_TX
MCLK_RX
MCLK_TX
15_36MHZ
DLL
EN_DLL
FLPAIR
APPENDIX A - Figure 1
Xilinx Corporation
2100 Logic Drive
San Jose, CA 95124
Date Last Modified: 1/9/4
Project: SBUSSYNC
Macro: SBUS_DLL
Date: 8/19/01



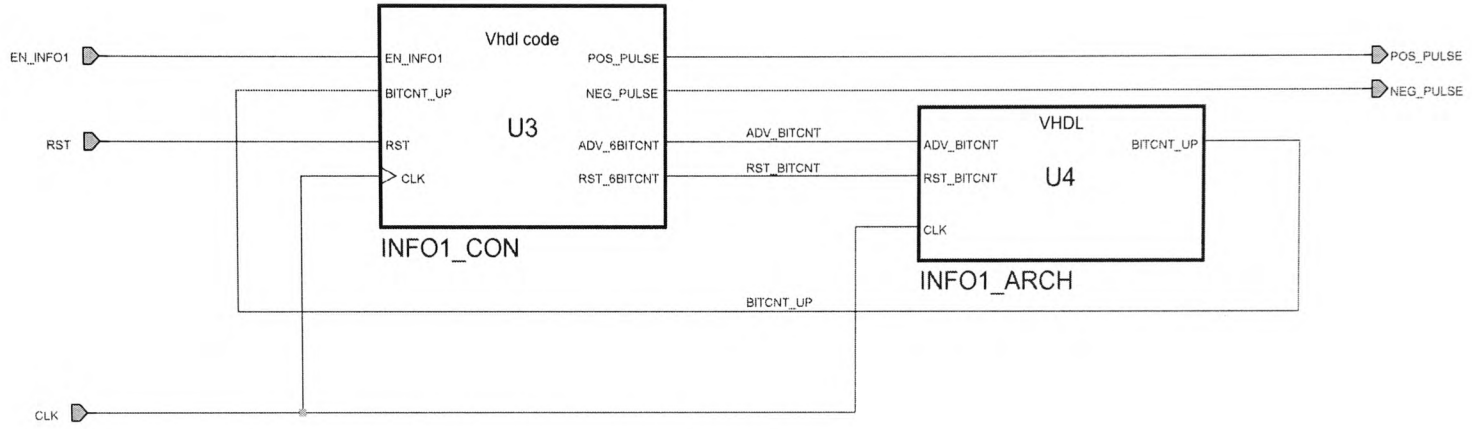
APPENDIX A - Figure 2

Xilinx Corporation	Project: SBUSSYNC
2100 Logic Drive	Macro: DLL
San Jose, CA 95124	Date: 8/19/01
Date Last Modified: 1/9/4	



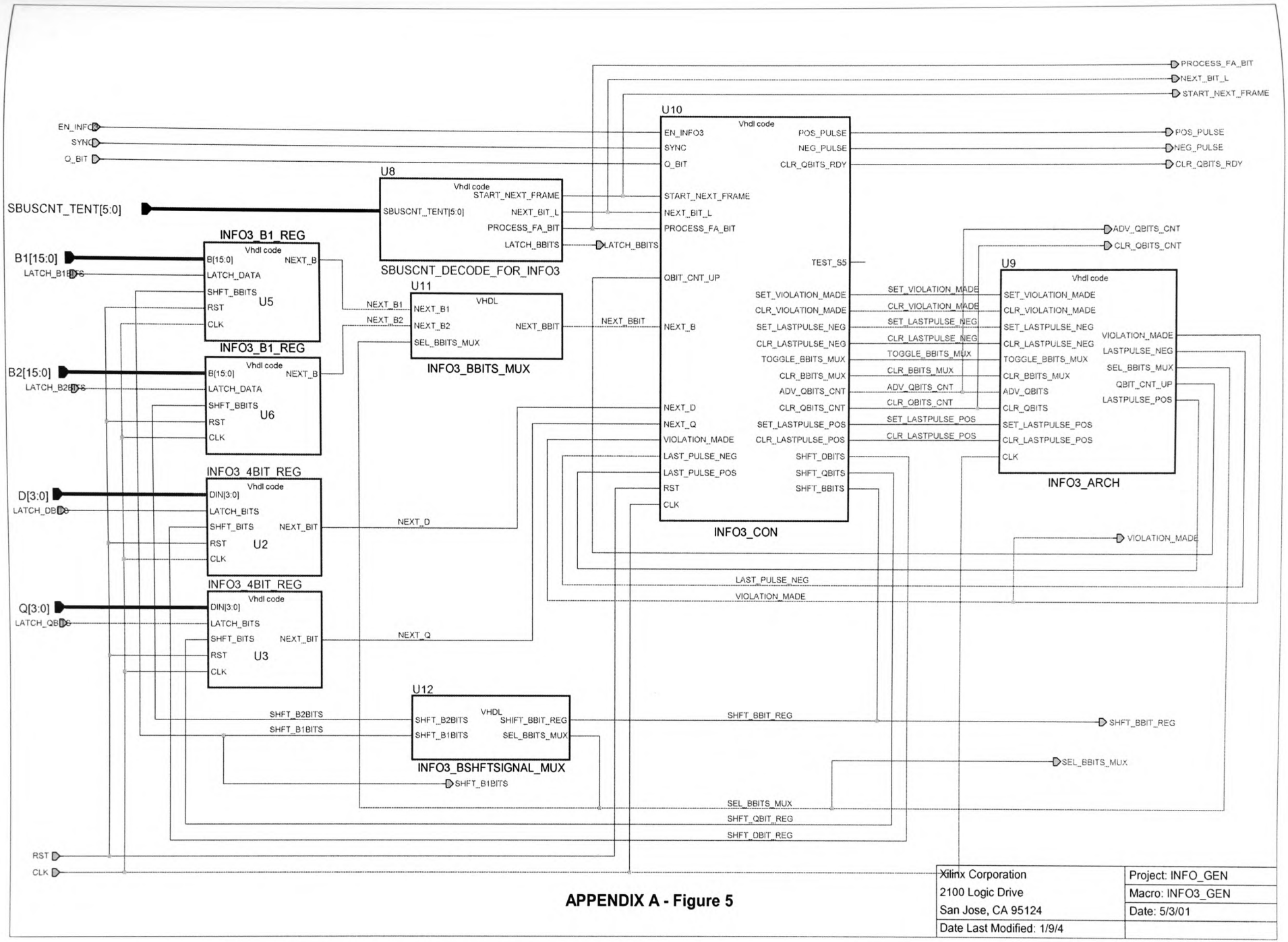
APPENDIX A - Figure 3

Xilinx Corporation	Project: INFO_GEN
2100 Logic Drive	Macro: INFOGEN_TETONT
San Jose, CA 95124	Date: 5/3/01
Date Last Modified: 1/9/4	



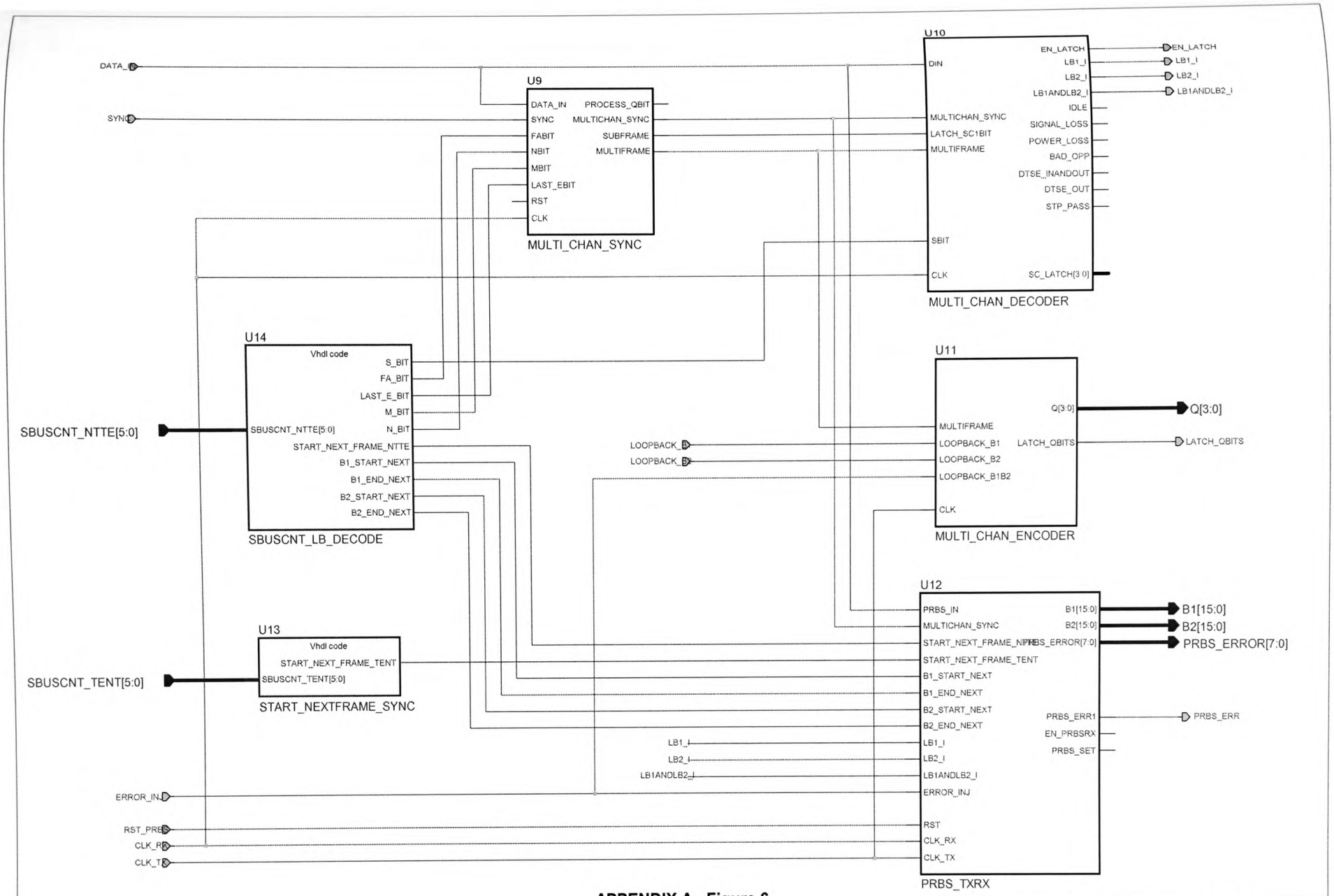
APPENDIX A - Figure 4

Xilinx Corporation	Project: INFO_GEN
2100 Logic Drive	Macro: INFO1_GEN
San Jose, CA 95124	Date: 5/3/01
Date Last Modified: 1/9/4	



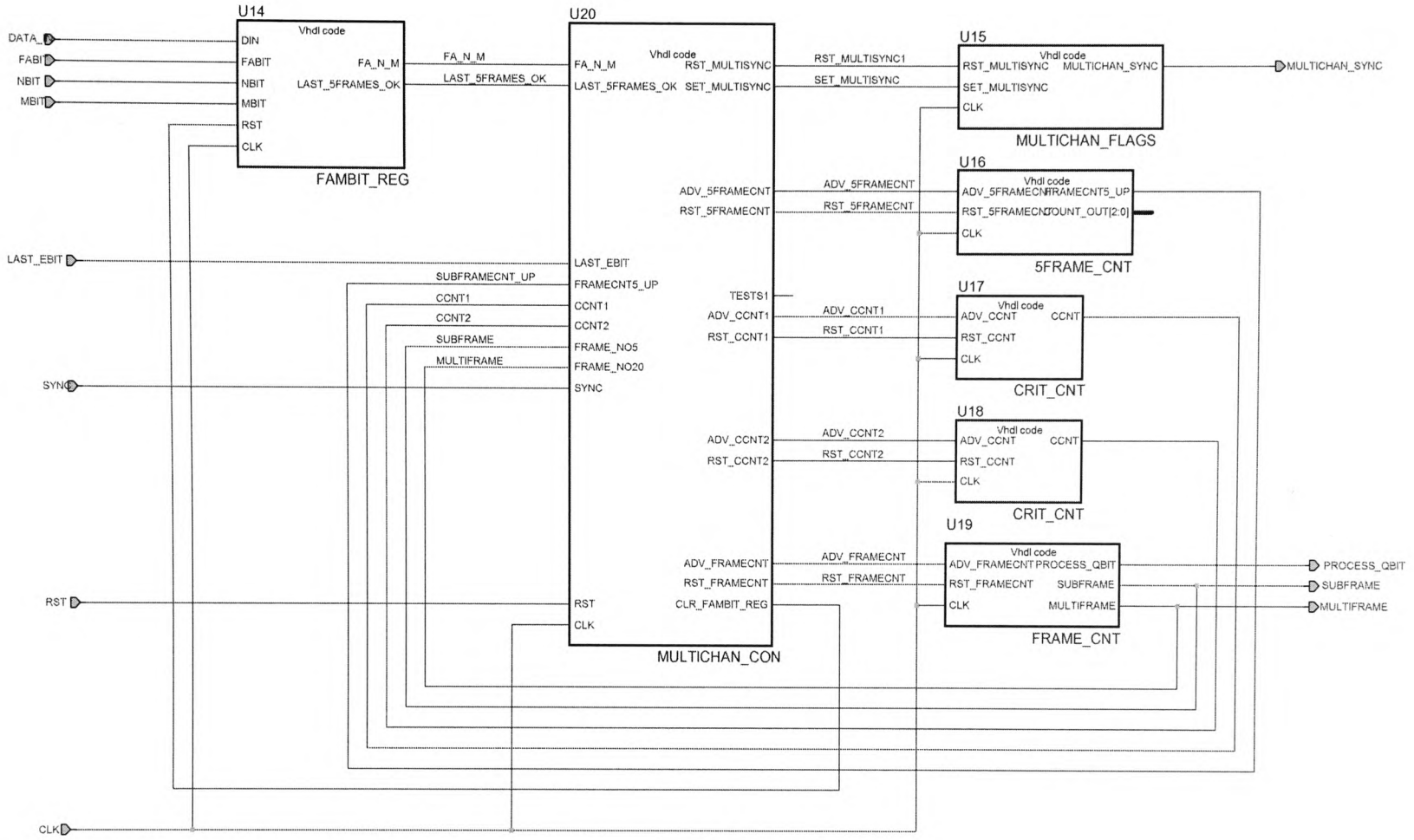
APPENDIX A - Figure 5

Xilinx Corporation	Project: INFO_GEN
2100 Logic Drive	Macro: INFO3_GEN
San Jose, CA 95124	Date: 5/3/01
Date Last Modified: 1/9/4	



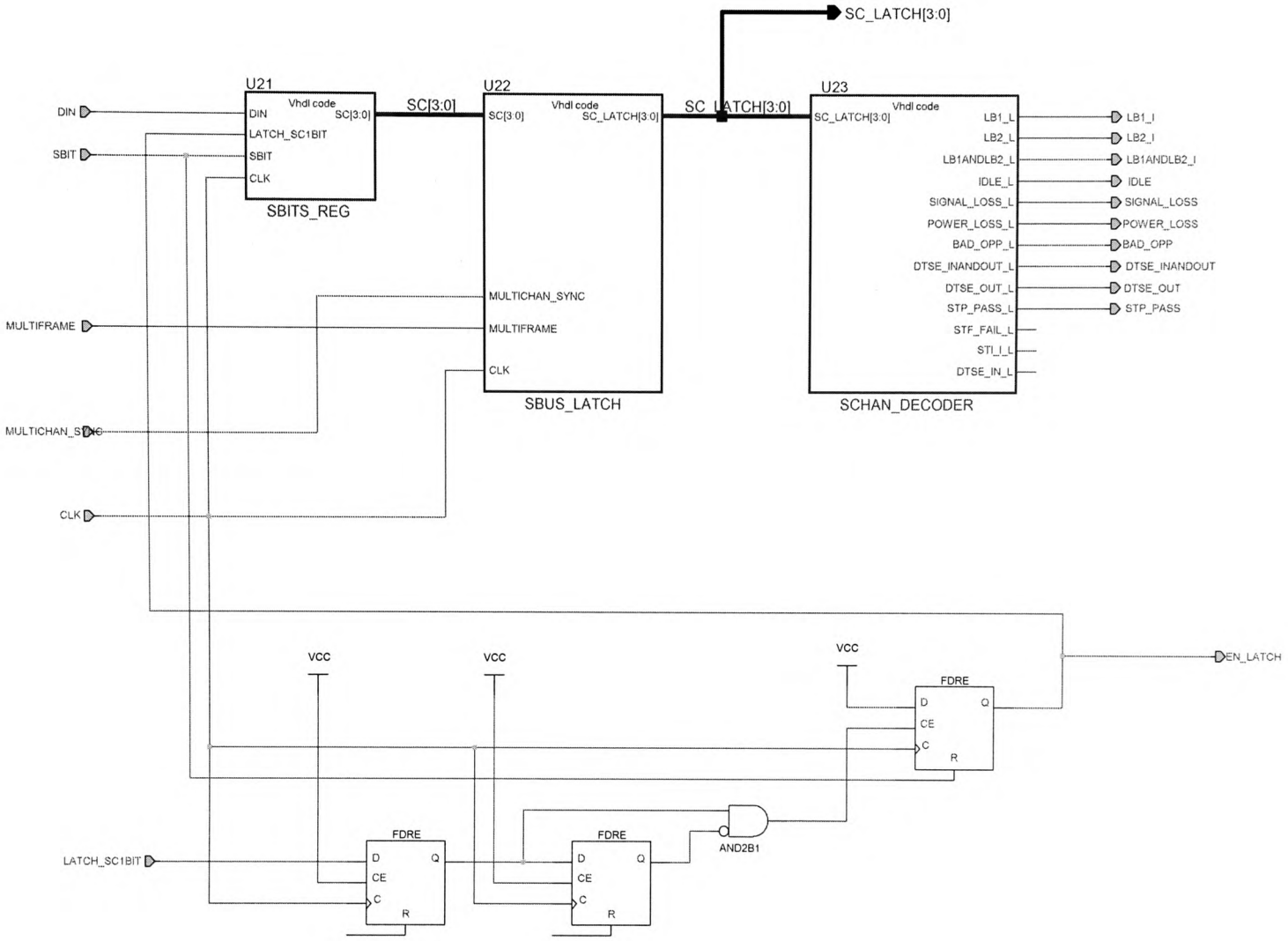
APPENDIX A - Figure 6

Xilinx Corporation	Project: DPLL_RX
2100 Logic Drive	Macro: LOOPBACK
San Jose, CA 95124	Date: 12/18/02
Date Last Modified: 1/9/4	



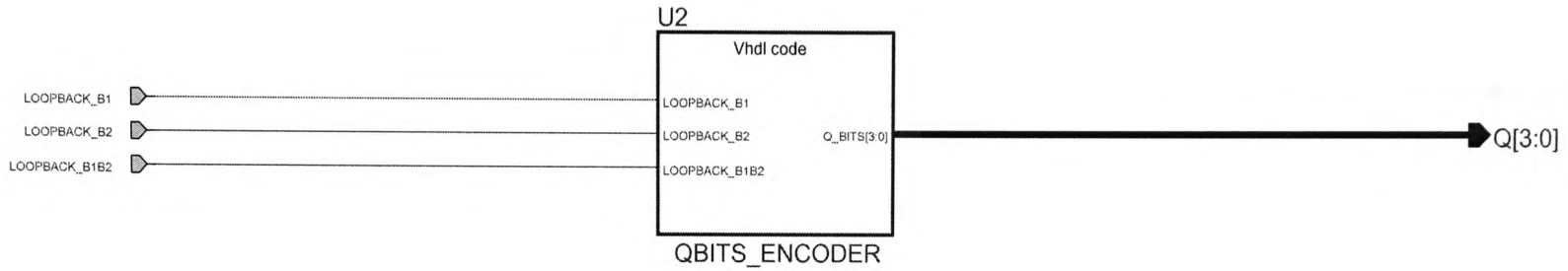
APPENDIX A - Figure 7

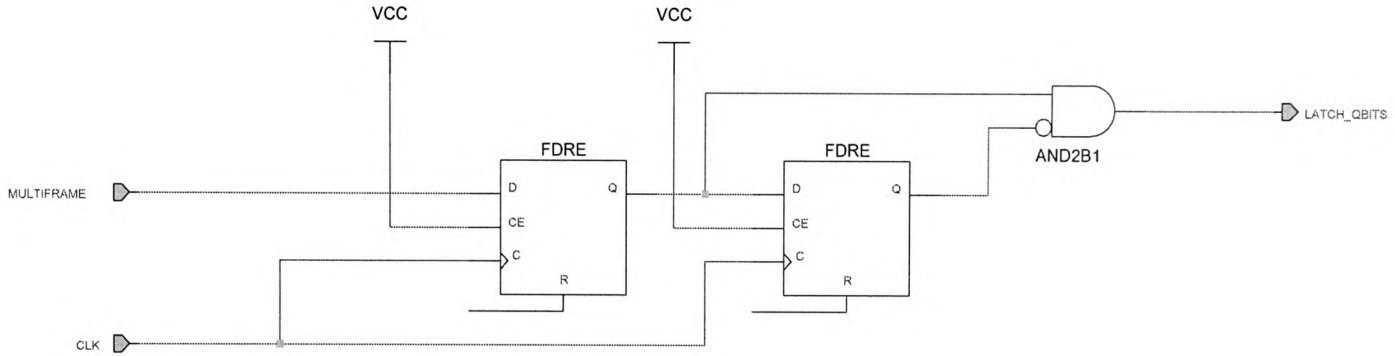
Xilinx Corporation	Project: SBUSTEST
2100 Logic Drive	Macro: MULTI_CHAN_SYNC
San Jose, CA 95124	Date: 12/18/00
Date Last Modified: 1/9/4	



APPENDIX A - Figure 8

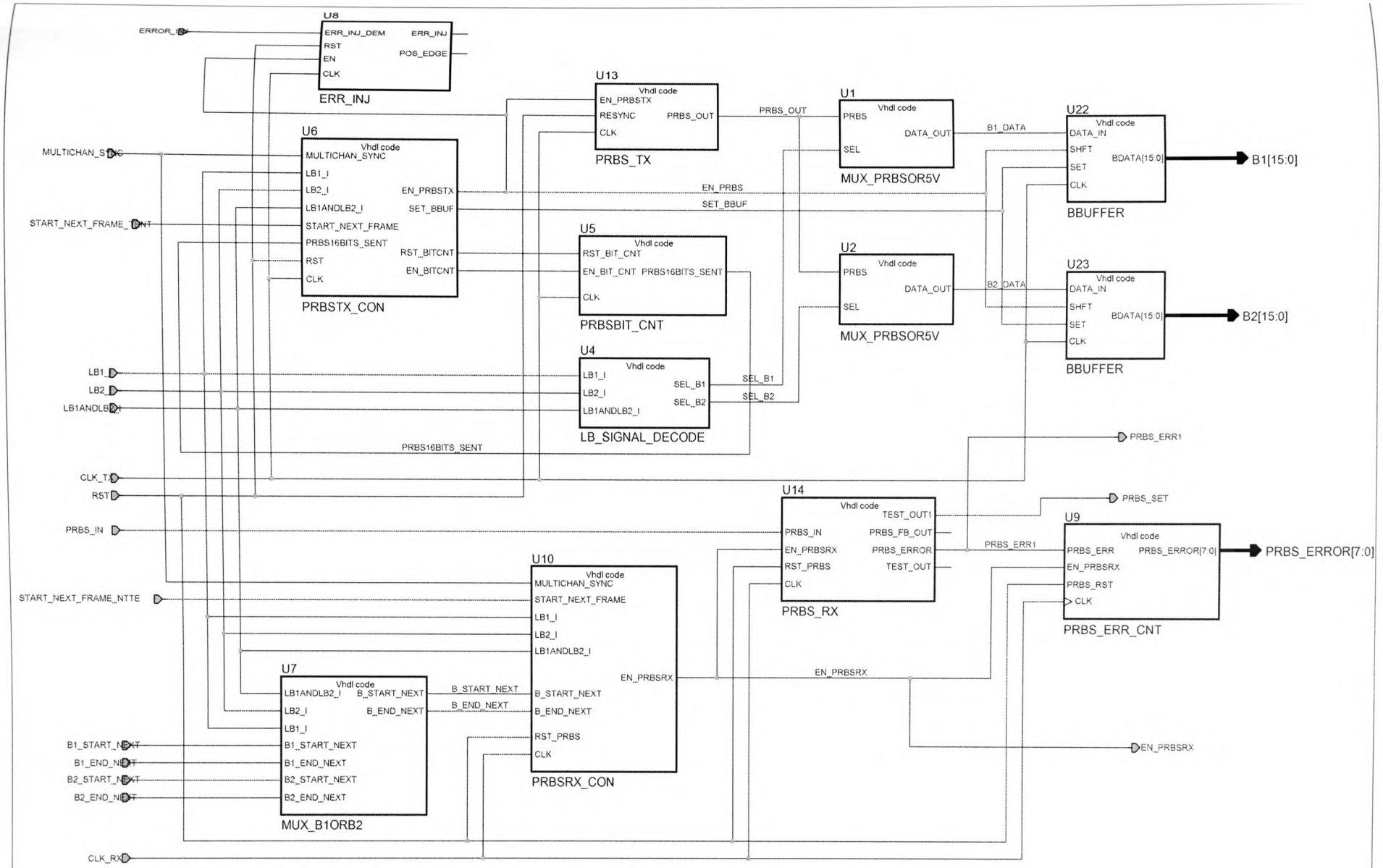
Xilinx Corporation	Project: SBUSTEST
2100 Logic Drive	Macro: MULTI_CHAN_DECODER
San Jose, CA 95124	Date: 12/18/00
Date Last Modified: 1/9/4	





APPENDIX A - Figure 9

Xilinx Corporation	Project: SBUSTEST
2100 Logic Drive	Macro: MULTI_CHAN_ENCODER
San Jose, CA 95124	Date: 12/18/00
Date Last Modified: 1/9/4	

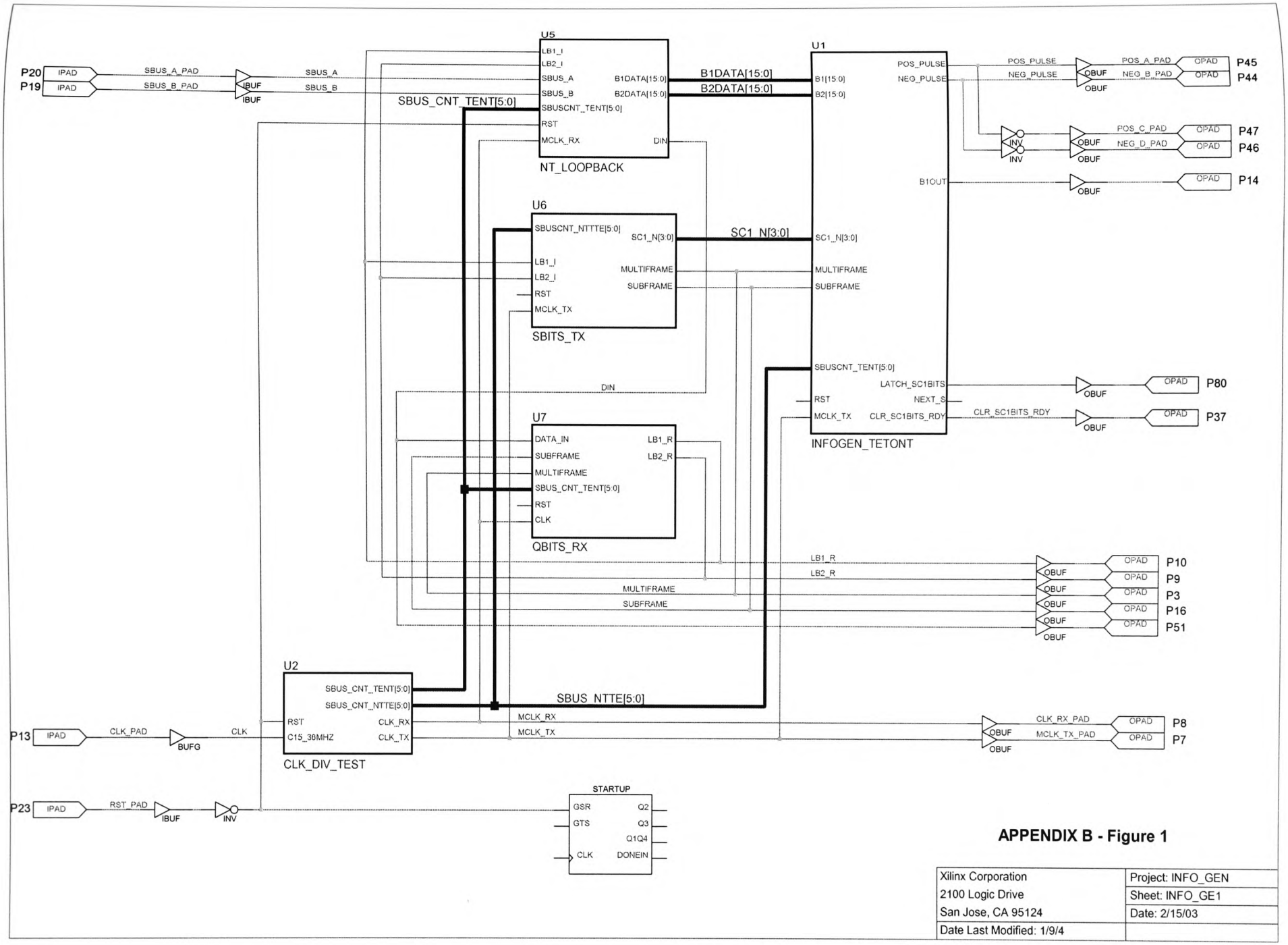


APPENDIX A - Figure 10

Xilinx Corporation	Project: SBUSTEST
2100 Logic Drive	Macro: PRBS_TXRX
San Jose, CA 95124	Date: 12/18/00
Date Last Modified: 1/9/4	

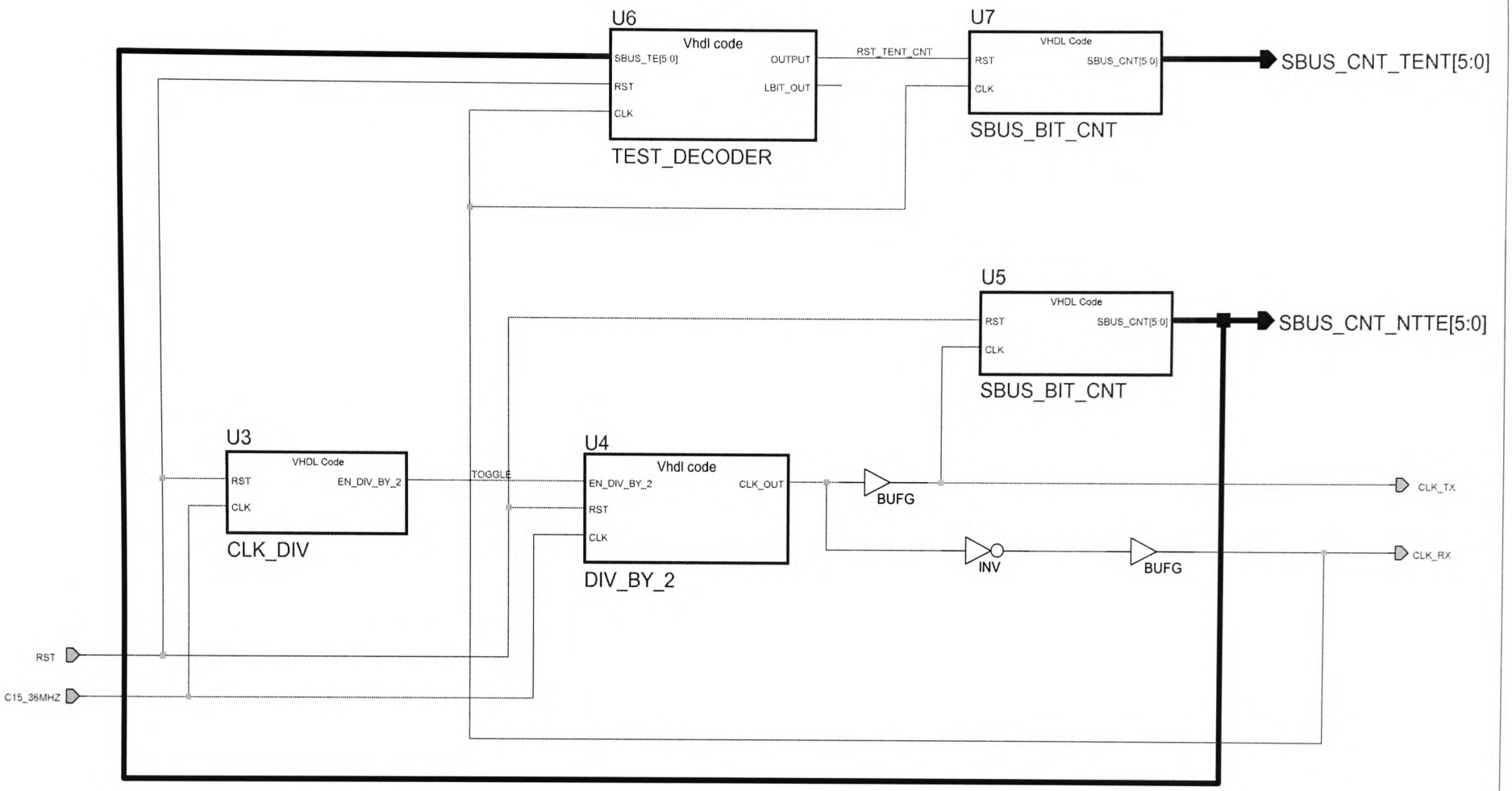
Appendix B.

The FPGA firmware block diagram for the NT.



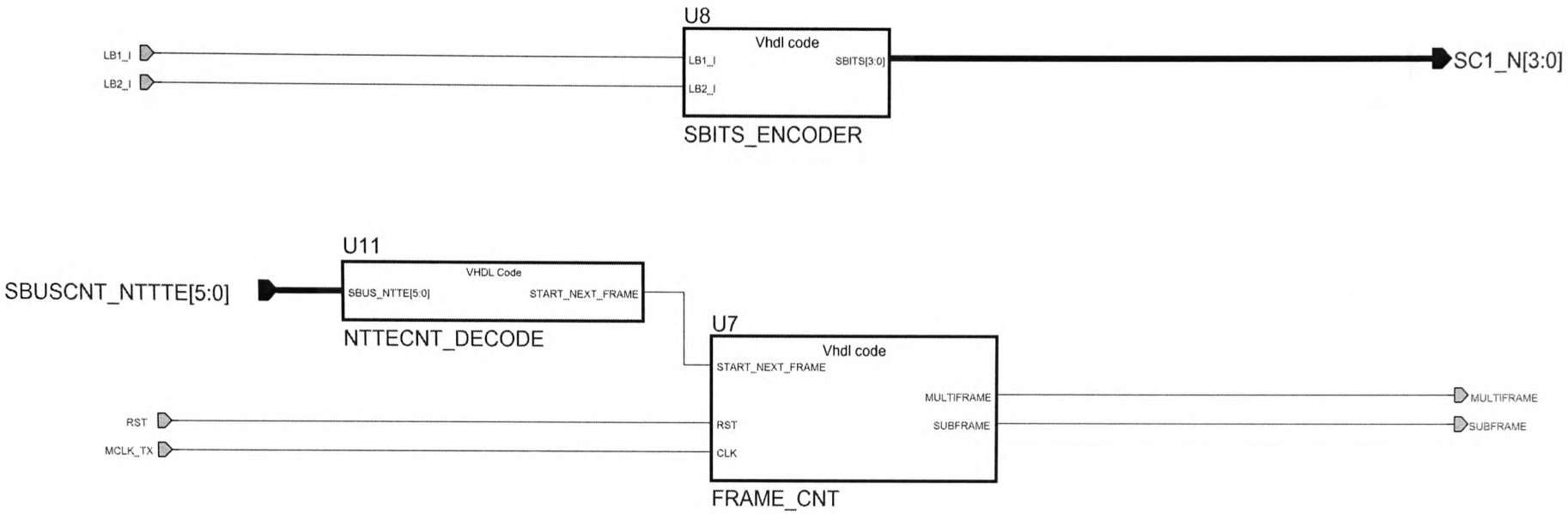
APPENDIX B - Figure 1

Xilinx Corporation	Project: INFO_GEN
2100 Logic Drive	Sheet: INFO_GE1
San Jose, CA 95124	Date: 2/15/03
Date Last Modified: 1/9/4	



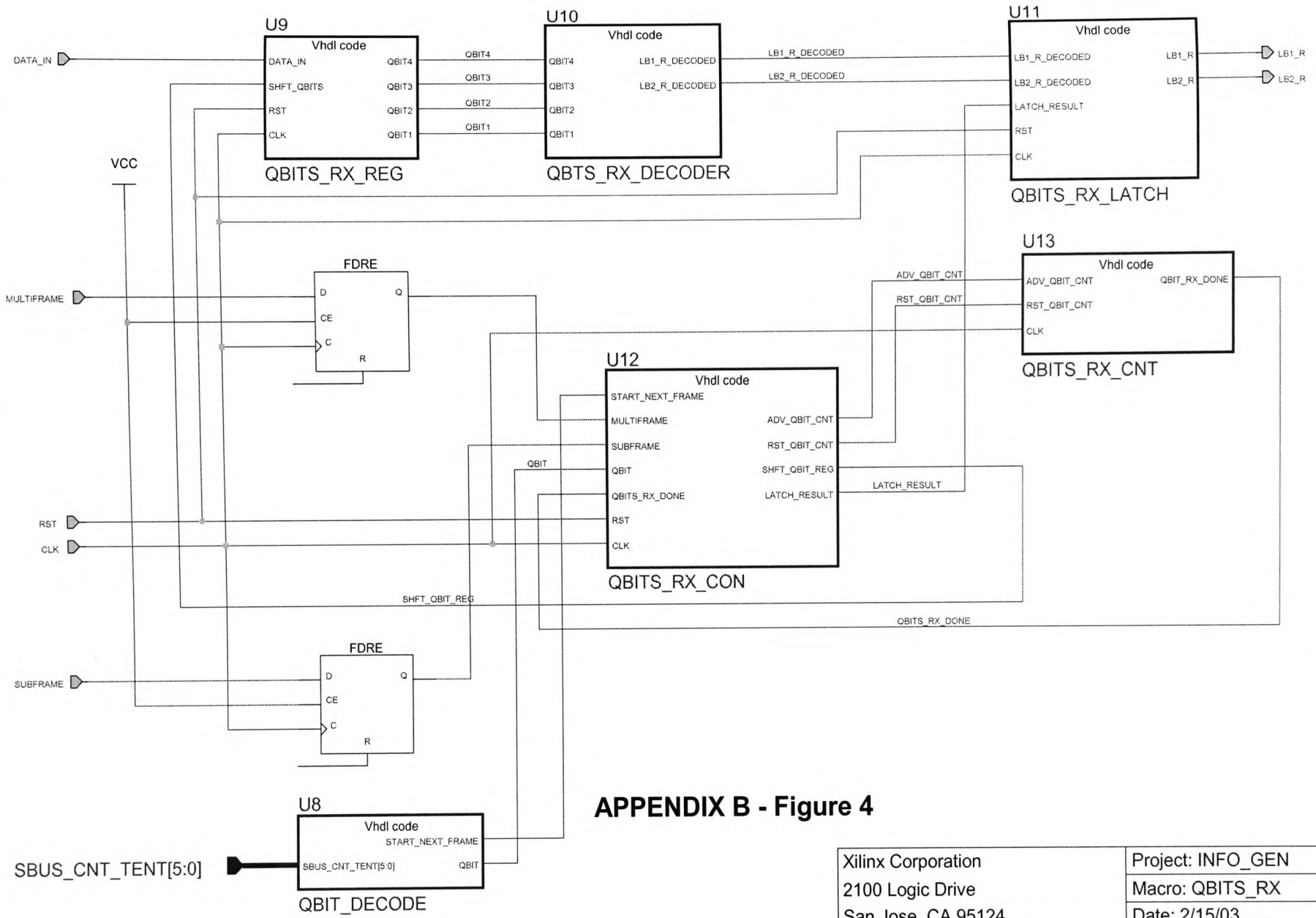
APPENDIX B - Figure 2

Xilinx Corporation	Project: INFO_GEN
2100 Logic Drive	Macro: CLK_DIV_TEST
San Jose, CA 95124	Date: 2/15/03
Date Last Modified: 1/9/4	



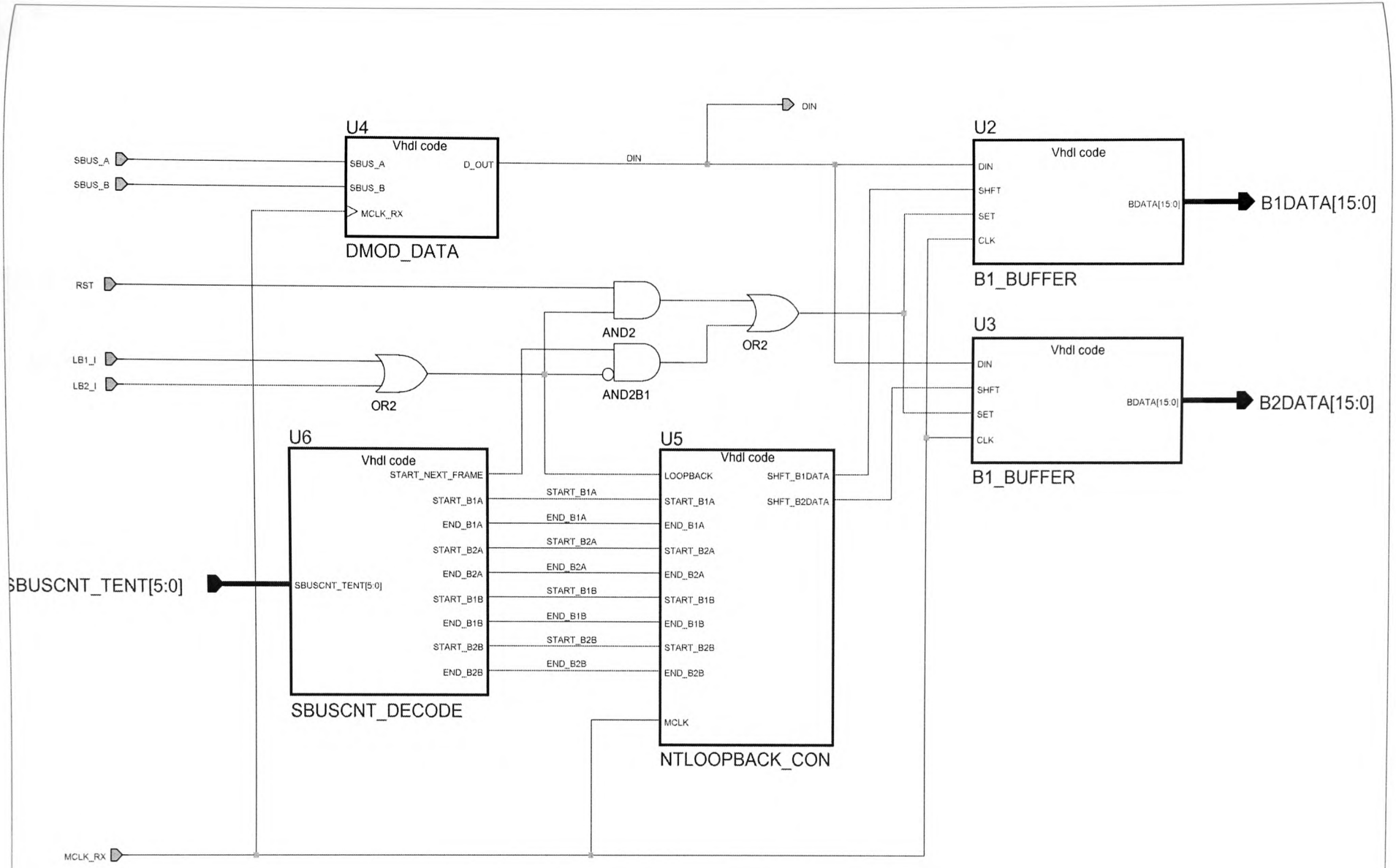
APPENDIX B - Figure 3

Xilinx Corporation	Project: INFO_GEN
2100 Logic Drive	Macro: SBITS_TX
San Jose, CA 95124	Date: 2/15/03
Date Last Modified: 1/9/4	



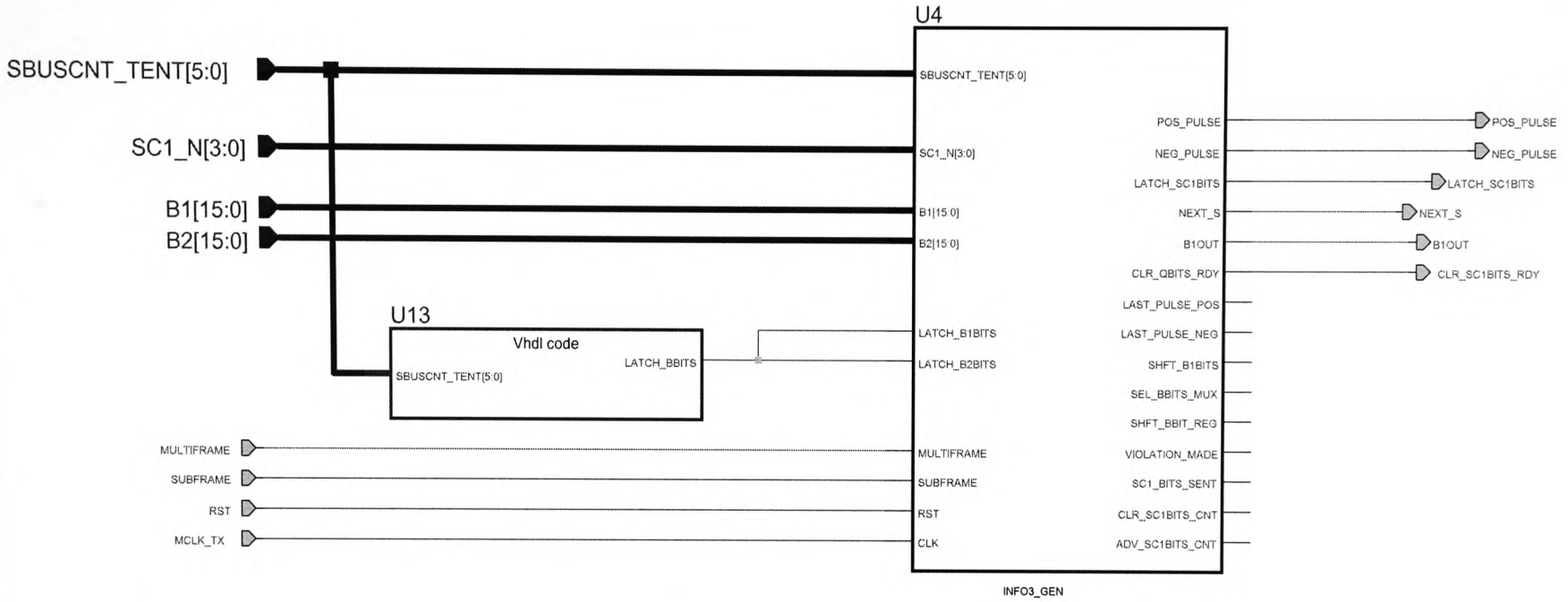
APPENDIX B - Figure 4

Xilinx Corporation	Project: INFO_GEN
2100 Logic Drive	Macro: QBITS_RX
San Jose, CA 95124	Date: 2/15/03
Date Last Modified: 1/9/4	



APPENDIX B - Figure 5

Xilinx Corporation	Project: NT_TEST
2100 Logic Drive	Macro: NT_LOOPBACK
San Jose, CA 95124	Date: 9/8/01
Date Last Modified: 1/9/4	



APPENDIX B - Figure 6

Xilinx Corporation	Project: INFO_GEN
2100 Logic Drive	Macro: INFOGEN_TETONT
San Jose, CA 95124	Date: 5/3/01
Date Last Modified: 1/9/4	

Report Number 3.

The application of Field Programmable Gate Array technology to the development of Basic-Rate ISDN monitoring systems.

Patrick Sugrue.

Report number 3 is part of a portfolio of projects and is submitted in partial fulfillment of the requirements of the University of Glamorgan for the degree of Master of Philosophy.

15 – December – 2003

Contents

Appendix

List of figures.

List of tables.

Glossary.

abbreviations

AMI	Alternate Mark Inversion
ASSP	Application specific standard products

ASIC	Application Specific Integrated Circuits
BERT	Bit Error Rate Test.
CAD	Computer Aided Engineering
DUT	Device Under Test
FPGA	Field Programmable Gate Array
HDLC	Higher Level Data Link Controller.
ISDN	Integrated Systems Digital Network.
IOM-2	ISDN Orientated Modular Interface – Revision two.
ITU-T	International Telecommunication Union – Telecommunications.
kbps	Kilo-bits-per-second.
LAPD	Link Access Protocol for the D channel.
NT	Network Terminator.
OSI	Open Systems Interconnect
PLL	Phase Locked Loop
S Bus	Subscriber bus.
TE	Terminal Equipment.
VHDL	Very High Speed Integrated Circuit Hardware Description Language

Chapter 1.

1. Introduction.

When a subscriber requests a Basic-Rate Integrated Systems Digital Network (ISDN) connection, a Network Terminator (NT) is first installed and commissioned. In Europe, the NT is installed and commissioned by the Telephone company and remains the property of the Telephone Company. In North America the subscriber may have access to the U interface point. In this case the responsibility for the installation and maintenance of the NT lies with the subscriber. In either case, the subscriber has access to 144 kilo-bits-per-second (kbps) available from the telephone company's exchange via the NT at the S interface point. The point at which the subscriber connects to the NT is designated the S interface point. The subscriber may connect a Terminal Equipment (TE) to the NT via a four-wire bus. A TE may be any ISDN compatible equipment. The four wires that connect the NT to the TE are collectively called the subscriber bus (S Bus). Differential signalling is employed on the S Bus and figure 1 demonstrates how the signalling flow is organised.

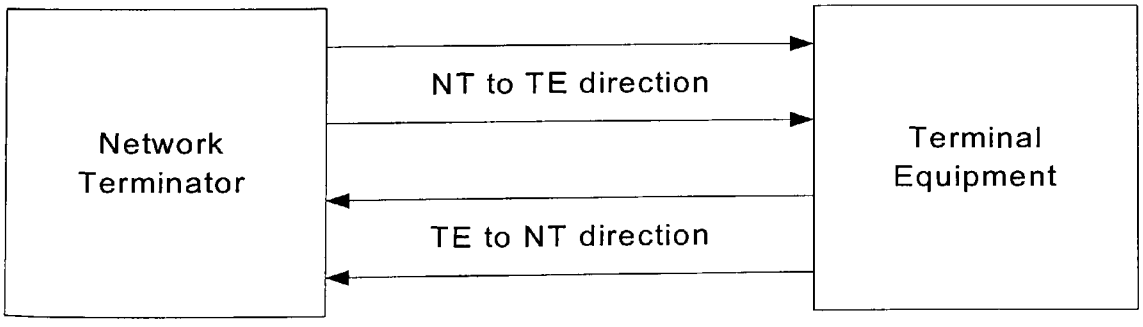


Figure 1 – The direction of data flow on the S Bus.

Data transfer on the S Bus is unidirectional. A twisted pair is employed for each direction of data flow. Data transmitted from the NT toward the TE is referred to as the NT to TE direction. Data transmitted from the TE toward the NT is referred to as the TE to NT direction. This labelling convention is used throughout this report. There are two basic types of connections that the subscriber may make when physically connecting to the S Bus. The subscriber may connect a single TE to the NT to make a point-to-point connection. Alternatively, the connection may be configured as a short passive bus or an extended passive bus where eight TEs are connected to the NT. The technical detail for the different ISDN wiring specifications is outlined in recommendation I.430 [1]. The complexity involved in ISDN wiring can often lead to wiring faults during installation. The S bus wiring polarity must be maintained for the entire cabling run. Any wiring errors will prevent the system from working properly. This study does not address directly how to analyse basic rate ISDN wiring problems. Companies engaged in the design of Telecommunications test equipment have already solved this problem adequately. The focus of the study is to improve on currently adopted hardware implementation methods that provide ISDN protocol analysis of telephone call set-up procedures.

The telephone call set-up procedure adopted for ISDN.

The Basic-Rate ISDN consists of two B channels used for data or voice calls operating at 64kbps and one D channel operating at 16kbps. The D channel is used to facilitate telephone call set-up. Both voice and data services will be referred to as telephone calls within the context of this report. The communication protocol used on the D channel is called Link Access Protocol for the D channel (LAPD). LAPD is a

modified form of the Higher Level Data Link Controller (HDLC) protocol. The Open Systems Interconnect (OSI) model may be used to categorise each signal carried on LAPD into one of three layers depending on the signals function. Dunlop has explained the seven layers of the OSI model in [2]. For ISDN systems only layers one to three are applicable. There are a number of stages involved in setting up an ISDN telephone call. The ISDN telephone call set-up procedure begins when a TE sends a signal from layer three to the layer one instructing layer one to establish synchronisation with the NT. Once synchronisation is established the physical layer signalling is active and call set-up signals may be transmitted on the D channel in both directions of transmission. The NT echoes the D channel information flowing in the TE to NT direction back to all the TEs in the network. These bits are referred to as echo bits or E bits and are transmitted in the echo channel. A TE will transmit a unique code toward the NT in the D channel. Only one TE may have access to the D channel at any one time. A TEs connected to the network monitors the echo channel from the NT and begins transmission on the D channel toward the NT when it recognises its own unique code. This way the NT controls which TE has access to the D channel at any one time. The D channel access procedure is explained in detail in Recommendation I.430. It is possible therefore to extract the D channel information for both directions of transmission by monitoring the D and E bits within the data frames flowing in the NT to TE direction only. This arrangement can be used to simplify the monitor design.

Figure 2 shows how the ISDN protocol analyser is connected to the S Bus. The Protocol analyser is transformer coupled to the S Bus and monitors the signals flowing between the NT and the TE. Should a call set-up fault occur on the S Bus, a protocol analyser may be employed to identify the problem. The analyser monitors

the signals associated with each OSI layer to allow detailed trouble shooting of the fault.

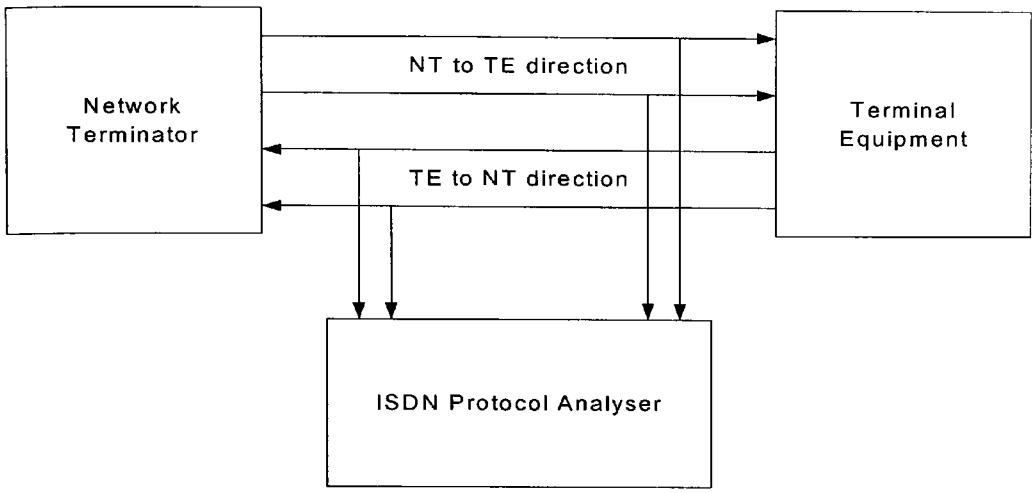


Figure 2 – The ISDN Protocol Analyser connected to the S Bus.

1.1 The weakness of currently adopted implementation methods.

Figure 3 presents a system block diagram identifying the hardware components of a typical ISDN protocol analyser. The diagram identifies three functional blocks. The Analogue Interface performs the analogue to digital conversion and is physically connected to the S Bus via an RJ-45 connector. The S Bus signals flowing in the NT to TE direction are tapped of pins four and five of the RJ-45 connector (RJ45A_PIN4, RJ45A_PIN5). The TE to NT data stream is connected to pins three and six of the same RJ-45 connector (RJ45A_PIN3, RJ45A_PIN6). These physical connections are standard for basic rate ISDN and are defined in Recommendation I.430.

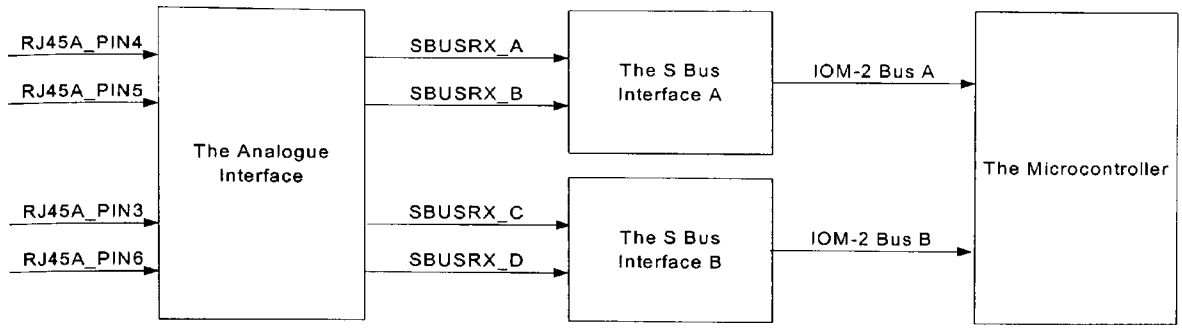


Figure 3 – A typical Protocol Analyser system block diagram.

The protocol analyser is required to monitor and identify layer one signals flowing on the S Bus. Layer one signals are defined in Recommendation I.430 as INFO0, INFO1, INFO2, INFO3 and INFO4 signals. The protocol analyser is also required to analyse layer two signals. Layer two and layer three signals are transmitted on the D channel of the S Bus. Layer two signals are defined in the International Telecommunication Union – Telecommunications (ITU-T) Recommendation Q.921 [3] and layer three signals are defined in Recommendation Q.931 [4]. In order to monitor both directions of transmission on the S Bus independently, two S Bus interface circuits are required as shown in the block diagram of figure 3. One monitors the signals flowing in the TE to NT direction and the other monitors the NT to TE direction. The Analogue Interface is transformer coupled to the S Bus so that the signal monitoring process does not interfere with the NT and TE operation. The term passive monitor is often used by manufactures in their product literature to describe this connection. Typical implementations employ commercially available Application specific standard products (ASSP) designed to interface with the ISDN S Bus. Typical ISDN S Bus Interface ASSPs include the Infineon PEB3081 and Motorola MC14557. The design features for each ASSP are detailed in the data sheets referenced in [5] and [6]. Theses ASSPs were designed to receive and transmit data simultaneously and are thus not readily suited to the

protocol monitoring application already outlined. Two S Bus interface ASSPs are required for this application. The receiver inputs of one ASSP are connected via a transformer to the NT to TE direction of the S Bus while the receiver inputs of the second ASSP are connected to the TE to NT direction. The problems with this approach are high economical cost and low design flexibility. The high cost is due primarily to the need for two ASSPs. The extra features offered by the ISDN ASSP that are not required by the application can be considered as wasted expenditure. It is recognised that ASSP manufacturers cannot provide a solution for every application. The resulting ASSP design solution is typically limited to features demanded by the high volume market leaving low volume applications unsatisfied. Telecom Test Equipment Designers usually employ commercially available ASSPs in an attempt to reduce the product development time demanded by the market. Designers begin by identifying suitable ASSPs that provide the required features. In this case ASSPs providing S Bus interfacing facilities are identified and then a suitable chip is selected. Attention is then focused on identifying a suitable microcontroller. The microcontroller to ASSP interfacing method is dictated by the ASSP design and this in turn limits the choice of microcontroller. For example, should the Infineon PEB2080 ISDN S Bus Interface chip be selected, a suitable microcontroller would be required to support the ISDN Orientated Modular Interface – Revision two (IOM-2) bus interface invented by Siemens and defined in [7]. The IOM-2 bus is an Industry standard bus for Interfacing ISDN chipsets. The Motorola 68302 microcontroller is a widely adopted solution for Basic Rate ISDN applications. Its internal HDLC controllers and IOM-2 interface make it a suitable choice in this case. The S Bus interface ASSPs identify the active layer one signals on the S Bus and can be configured to extract the D channel data from both directions of transmission. This

information is passed to the microcontroller via the IOM-2 bus for analysis. This is one example of forced pairing between an ASSP and Microcontroller that highlights the limited implementation options available to designers who choose to develop systems with commercially available ASSPs. These ISDN ASSPs are original in their design and it is unusual to find a second manufacturer providing an interchangeable product. This means that ASSP procurement problems may undermine production schedules. Figure 4 proposes a more efficient method to implement an ISDN protocol monitor. In this case the ASSPs are replaced with a single field programmable gate array (FPGA).

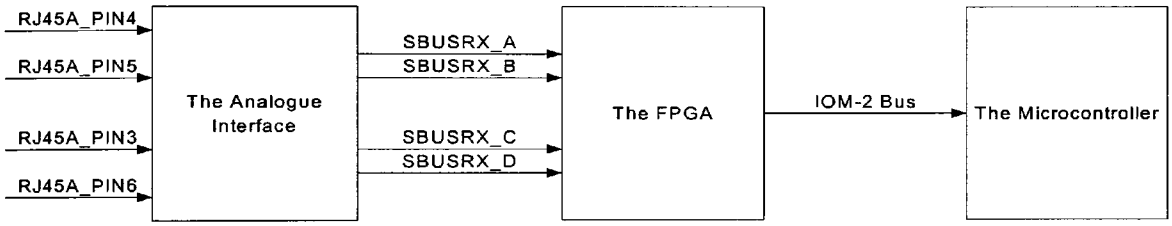


Figure 4 – The improved Protocol Analyser system block diagram.

1.2 The objectives of the study.

The following objectives have been identified for project number 3.

1. To develop an ISDN protocol monitoring method with FPGA technology that reduces the cost of the system and the dependence on single ASSP manufacturers while increasing design flexibility to satisfy the demands of a changing market.
2. To implement a solution that allows the reuse of FPGA circuitry developed for project number 2 to increase development time and reduce time to market.

3. To extend the analogue electronics facilities of the hardware platform to allow integration of the protocol monitor feature.

1.3 The achievements.

The FPGA Firmware design exercise successfully produced the interfacing functionality required to identify the layer one signals on the S Bus and extract D channel data for onward transmission via the IOM-2 Bus to the Microcontroller. The design was successfully implemented using the Very High Speed Integrated Circuit Hardware Description Language (VHDL) and was successfully simulated using a VHDL Test bench. The design was successfully placed and routed on a single Xilinx 4010XL/Spartan FPGA. A VHDL Test bench was designed and implemented to successfully simulate the Basic rate ISDN S Bus. The Test bench simulated all the required S Bus signals and provides designers with all the facilities required to test and validate ISDN Terminal Equipment designs.

1.4 Report structure.

Chapter two describes the Analogue Interface design. The analogue interface design is part of the contribution to knowledge and was designed along with the Firmware circuitry specifically for the ISDN application presented in this study. Timing diagrams are used to explain how the analogue circuit operates and reference is made to report number 2 where a detailed explanation of the design is presented.

Chapter three focuses on the FPGA Firmware design and forms the main contribution to knowledge presented by the study. An overview of the design is

presented and the operation of each component is described in detail. The layer one signal identification circuits, the D channel extraction circuitry and the IOM-2 Interface designs are explained. The implementation results of the FPGA place and route process are presented.

Chapter four details the test procedure used in the simulation of the Firmware design. An explanation of how the VHDL Test bench is designed to simulate the signals flowing in both directions of transmission on the S Bus is given. A test strategy is developed to allow the verification of each component in the Firmware design.

Chapter five presents the Test bench results. An in-depth analysis of the test results is performed and the Firmware design is verified and validated.

Chapter six presents the conclusions. The overall achievements are reviewed in relation to the objectives and suggestions for future work are made.

Chapter 2

2 The Analogue Hardware Design.

The Analogue design employed here is identical to that implemented for the ISDN Bit Error Rate Test (BERT) System already presented in report number two. The main difference is that the ISDN Monitor employs two analogue receiver circuits in order to facilitate signal monitoring in both directions of ISDN transmission. The Analogue receiver is shown in appendix A. The Analogue receiver circuit is transformer coupled to the ISDN line in the same way as that presented in the ISDN BERT system. The jumpers JP4 and JP5 disconnect the differential signal transmitter circuit used when the FPGA is configured to operate in Terminal Equipment mode. The capacitors C5, C6, C8 and C9 are not fitted and zero-ohm links are placed, as capacitive coupling is not used. The timing diagram for the analogue signal receiver circuit is shown in figure 5. The timing diagram shows both differential signals on the ISDN transmission line between the NT and the TE. The signal components have been represented separately for clarity. The signals RJ45A_PIN4 and RJ45A_PIN5 represent data flowing from the NT toward the TE. The signals RJ45A_PIN3 and RJ45A_PIN6 represent data flowing from the TE toward the NT. The data frames presented in the timing diagram have been reduced in detail to portray the synchronisation pulses only. The receiver output signals FPGA1_RXC and FPGA1_RXD provide a buffered version of the NT to TE direction of data transmission to the FPGA. The receiver output signals FPGA1_RXA and FPGA1_RXB buffer the TE to NT direction of data transmission. The buffered signals are connected to the FPGA and analysis of the data is performed. Chapter 2 of Report 2 details the precise operation of the analogue receiver circuit and presents test

results for one of the receivers. Both signal receivers are exact duplications of each other.

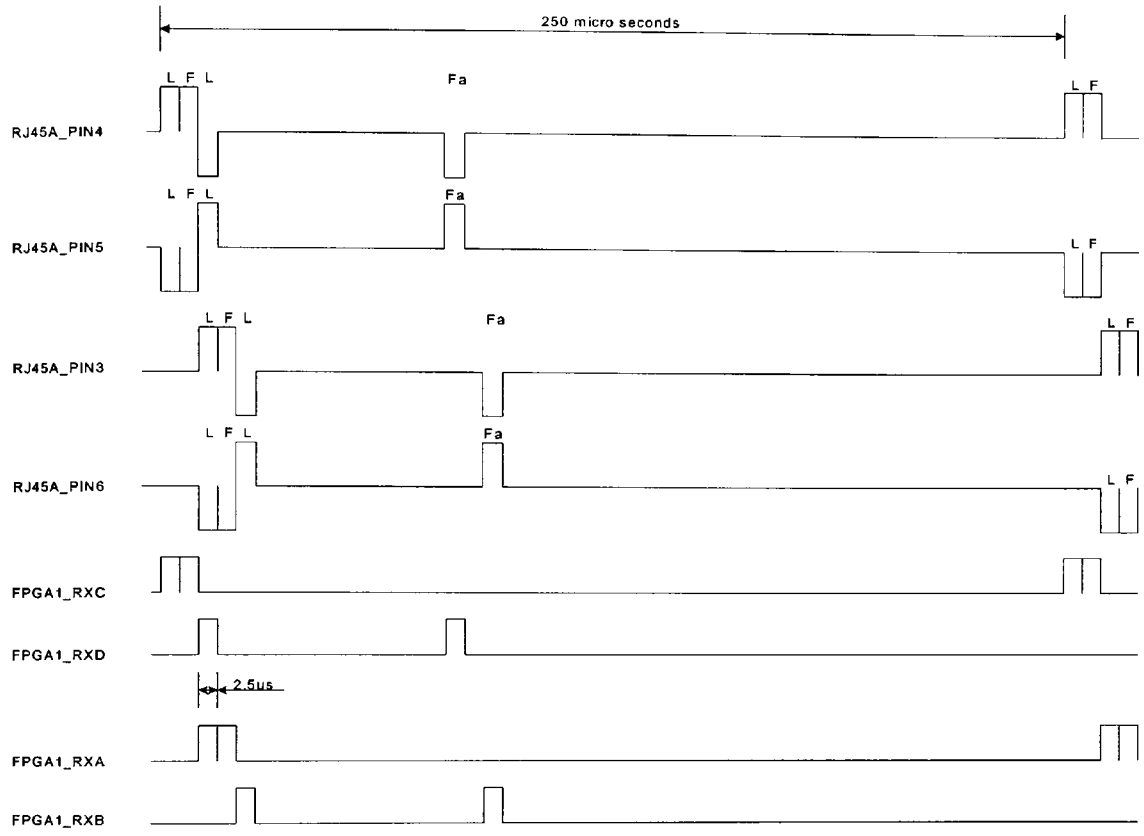


Figure 5 - The timing diagram for the analogue receiver circuit.

Chapter 3

3 The FPGA Firmware Design.

The FPGA Firmware provides the interfacing between the ISDN S Bus and the IOM-2 bus connected to the microcontroller. A simplified top-level block diagram for the functional hardware blocks in the Firmware design is presented in figure 6. A more detailed block diagram is presented in appendix B, figure 1. The FPGA inputs are on the left-hand side of the diagram and the FPGA outputs appear on the right-hand side. The 15.36MHz crystal oscillator is used as the master clock for the FPGA. The other clocks are synchronously derived from the master clock. In the following each functional block is discussed generally.

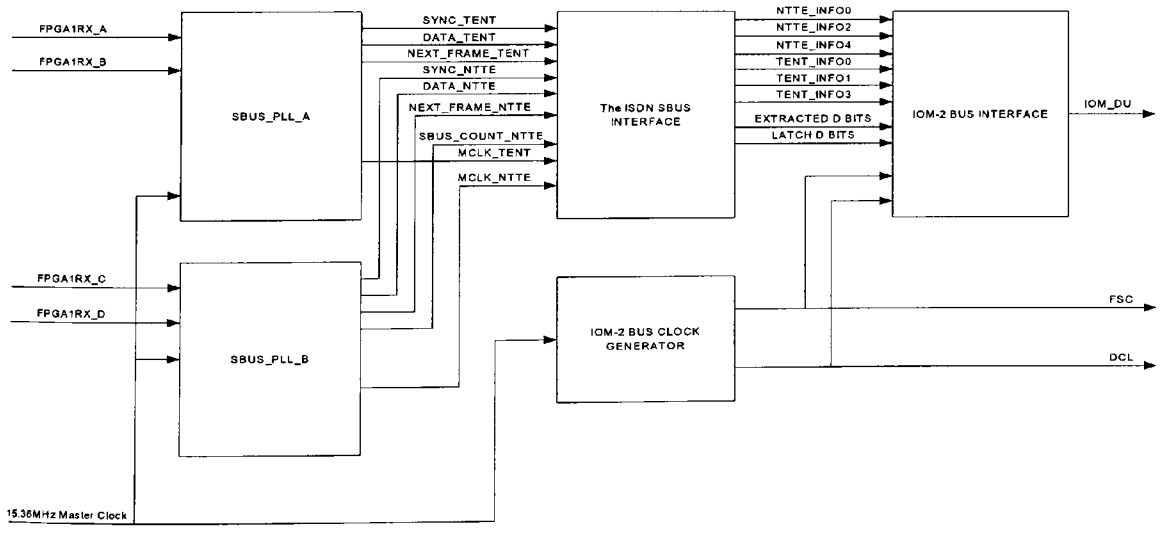


Figure 6 - The top-level block diagram for the ISDN Monitor Firmware.

The SBUS synchronisation and Phase Locked Loop (PLL) circuit (SBUS_PLL) already presented in Report 2 chapter 5 is used again here. The objective of the synchronisation and PLL circuit is to establish synchronisation with the ISDN S bus, enable phase locking of system clocks with the ISDN, and perform Alternate Mark Inversion (AMI) to binary conversion of the extracted S Bus data. The synchronisation and PLL circuit is used twice. One SBUS_PLL circuit is employed to establish synchronisation with the TE to NT direction of data transmission. This is labelled as SBUS_PLL_A on the block diagram. The other circuit is connected to the NT to TE direction of data transmission and is labelled as SBUS_PLL_B. Both circuits operate independently and when synchronisation is achieved the signals SYNC_TENT and SYNC_NTTE are set to logic one. In the synchronised state the PLL associated with both synchronisation circuits will ensure that the 192kHz S Bus sampling clocks (MCLK_TENT and MCLK_NTTE) are coincident with the centre point of each symbol on the S Bus. This ensures that data sampling occurs when the data on the S Bus is valid. The extracted S Bus data from both synchronisation circuits (DATA_TENT and DATA_NTTE) represent the binary data on the ISDN line. The data on the ISDN S Bus is organised into data frames. The S Bus frame in the NT to TE direction of data transmission contains both the D channel and E channel data. The S Bus count sequence (SBUS_COUNT_NTTE) produced by the SBUS_PLL_B circuit is used to identify each S Bus bit within the data frame. The count sequence is broadcast to the other blocks in the design to facilitate data bit extraction from the ISDN S Bus. The SBUS_PLL circuit has already been explained and detailed results confirming its operation have been presented in Report 2. It will not be dealt with further here.

The ISDN S Bus Interface performs the INFO signal identification and the D and E bits extraction procedure. The S Bus and the IOM-2 Bus operate at different clock rates. The E and D bits that are extracted from the S Bus are buffered within the ISDN S Bus Interface and transferred to the IOM-2 Bus Interface for onward transmission toward the Microcontroller. The synchronisation of the S Bus to the IOM-2 Bus can be explained with the aid of the timing diagram presented in figure 7. The timing diagram is not to scale but is sufficient to show the relationship between the S Bus and the IOM-2 Bus.

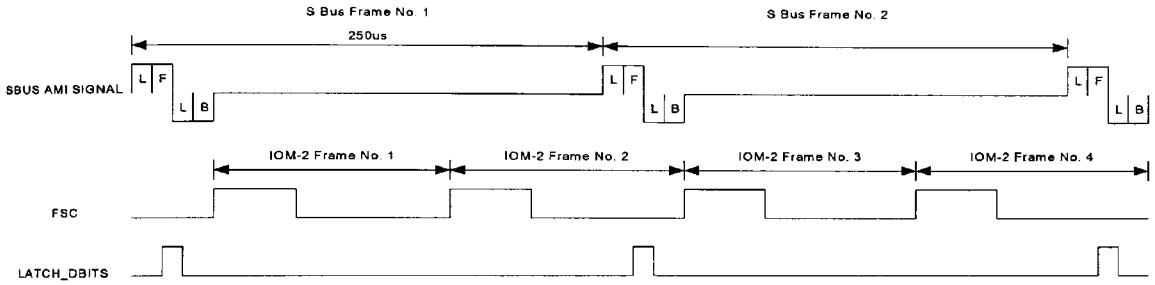


Figure 7 - The timing diagram showing the S Bus and IOM-2 Bus relationship.

The timing diagram presents two S Bus frames. The frame detail has been reduced to show the L, F, L symbols and the first bit of B channel data. The IOM-2 Clock Generator Block monitors the S Bus count sequence and derives the frame synchronisation clock (FSC) and the Data clock (DCL) for the IOM-2 Bus. The Data clock for the IOM-2 bus is omitted here but will be dealt with later in section 3.7. Once the S Bus synchronisation circuit (SBUS_PLL) has established synchronisation with the S Bus, the resulting S Bus count sequence is used to produce the FSC signal. The timing diagram in figure 7, shows the relationship between the S Bus AMI signals and the FSC clock. A detailed explanation of the FSC clock generation is provided in section 3.7. From the timing diagram it can be seen that there are two

IOM-2 frames sent within one S Bus frame. The eight bits of D and E channel information is extracted from the S Bus data frame and stored by the S Bus interface. The signal LATCH_DBITS occurs at the end of each S Bus frame and the eight bits of extracted D and E bits are loaded into the IOM-2 Bus interface. The IOM-2 Bus Interface inserts two bits of D channel data and two bits of E channel information into the IOM-2 data-up line (IOM_DU) per IOM-2 frame. The IOM-2 data-up line is a signal line on the IOM-2 bus and is the data transmission line from the FPGA to the Microcontroller. After two IOM-2 data frames have been sent to the Microcontroller all eight D and E bits will have been transmitted and the next active LATCH_DBITS signal loads the next eight bits to be processed. In this way the S Bus interface is extracting D and E channel information while the IOM-2 Bus Interface is transmitting D and E channel data on the IOM-2 Bus to the Microcontroller. The following sections discuss the operation of the ISDN S Bus Interface block, The IOM-2 Interface block and the IOM-2 Clock Generation Block.

3.1 The ISDN S Bus interface.

The ISDN S Bus interface (SBUS_DCHAN) performs the extraction of the E and D bits from the S Bus and maintains an indication of the currently active signals flowing between the NT and the TE. The top-level block diagram for the S Bus interface is shown in figure 2 of Appendix B. The S Bus count sequence decoder (DCHAN_DECODE_ARCH) and the D channel shift-register (DBITS_REG_ARCH) facilitate the E and D channel extraction process. The D and E channel extraction process samples the D and E bit data from an S Bus and stores the data within the S Bus Interface. This stored information is then sent to the IOM-2 Bus Interface on an

eight bit signal bus (DBITS) for onward transmission to the Microcontroller. The signals that may become active in the NT to TE direction are the INFO0, INFO2 and INFO4 signals. The signals that may become active in the TE to NT direction are the INFO0, INFO1 and INFO3 signals. Individual circuit blocks monitor the active signals on the S Bus. Each circuit is dedicated to the identification of one INFO signal. The individual circuit blocks operate in parallel and are continuously hunting for their respective signals. Once a signal is recognised an indication of the active signal is made. The design philosophy is to indicate the presence of a signal only when that signal has been unambiguously identified. An undefined signal exists until successful signal identification is made.

The INFO0 monitor (INFO0_MON), the INFO1 monitor (INFO1_MON) and the INFO signal decoder (TENT_INFO_CON) facilitates the identification of the INFO signals in the TE to NT direction of data transmission. An INFO0 will be the active signal when both the NT and TE are not transmitting. The INFO0 monitor identifies the presence of an INFO0 signal. Once a TE has been instructed to set up a telephone call, it transmits an INFO1 signal towards the NT. This brings the NT out of its power down mode. The NT then transmits an INFO3 towards the TE. Once synchronisation with the signal from the NT has been established it can be deduced that an INFO3 is active. The INFO3 condition can then be assumed when the SYNC signal is high. Once the INFO0 condition has elapsed and while SYNC is low an INFO1 may be active. The INFO1 monitor is required to identify this condition.

An INFO0 monitor circuit identical to that used to monitor the TE to NT direction is employed again to hunt for the presence of an INFO0 signal in the NT to TE direction. Once the S Bus Synchronisation and PLL circuit achieves synchronisation with the data frames flowing in the NT to TE direction, it sets the

signal SYNC_NTTE high. At this stage the INFO signal on the S Bus may be an INFO2 or an INFO4. An active INFO2 signal is determined by the binary value of the A bit associated with the NT to TE data frame. The INFO2 monitor (INFO2_MON) checks the A bit and indicates when an INFO2 is active on the line by setting the INFO2 signal high. If the SYNC_NTTE signal and the INFO2 signal are both high then it can be deduced that the active signal is an INFO2. If the INFO2 signal is low and the SYNC_NTTE signal is high it can be deduced that the active signal on the line is an INFO4.

3.1.1 The D channel extraction process.

The timing diagram for the data extraction process from both D channels is shown in figure 8. The S bus count sequence (SBUSCNT_NTTE) is derived from the S Bus Synchronisation and PLL circuit. Following the convention adopted throughout the project, the data transmitted from the NT to the TE is synchronised with the sampling clock (CLK_RX) to produce the data stream labelled as DATA_IN on the timing diagram. Once synchronisation with the NT to TE data frames has been established, the S bus count sequence can identify each E bit and D bit within the frame. When the S bus count sequence reads 9, 22, 33, 44 an E bit is active and the signals EBIT_A through EBIT_D indicate this condition. When the S bus count sequence reads 10, 23, 34, 45 a D bit is active and the signals DBIT_A through DBIT_D indicate this. Two hardware sub-blocks are used to facilitate the D and E channel extraction. The S bus Count Decoder (DCHAN_DECODE_ARCH) implements a binary decoder that produces the before mentioned signals. The D channel shift-register (DBITS_REG_ARCH) is clocked with the sampling clock and is used to latch the D and E channel data. The D channel and E channel active indications from the S bus

decoder enable the shift-register operation thus storing the D and E channel information synchronously with the sampling clock at the appropriate times. The Shift-register has provision to store the four bits of D channel data and four bits of E channel data. When a frame of data has been processed, the eight-bit parallel signal bus (DBITS) will hold the extracted D and E channel data. The S bus count decoder produces the signal LATCH_DBITS when the count sequence reads 46. The signal LATCH_DBITS synchronously latches the four D bits and four E bits into the IOM-2 bus interface for onward transmission to the Microcontroller.

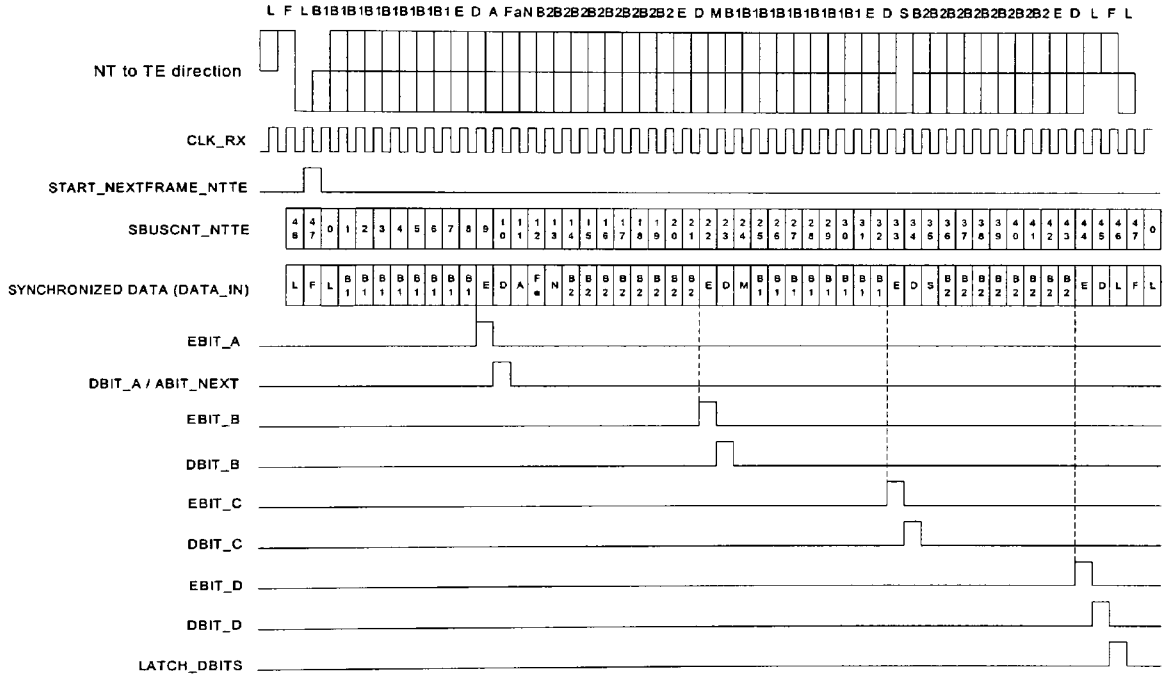


Figure 8 - The timing diagram for the D channel extraction process.

3.1.2 The INFO1 signal monitor.

The INFO1 signal monitor circuit (INFO1_MON) identifies the presence of an INFO1 signal on the telephone line. The INFO 1 signal is defined as a repetitive signal consisting of two consecutive binary zeros (marks) followed by six consecutive binary ones (spaces). The AMI signal as it appears on the telephone line is shown in figure 9.

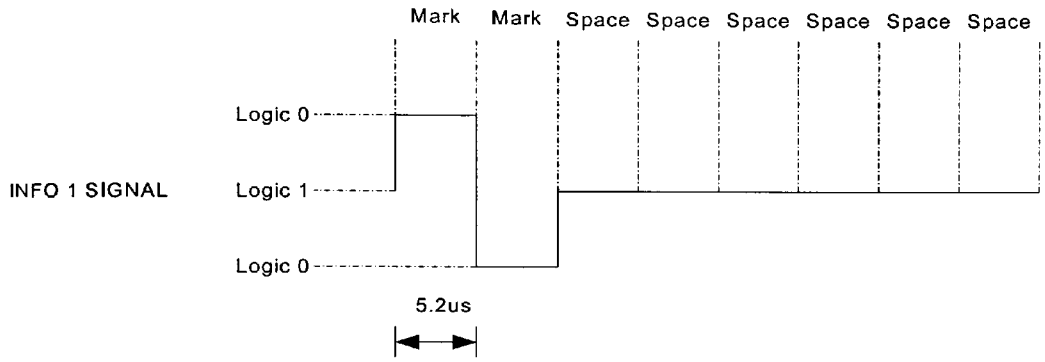


Figure 9 - The INFO1 signal.

The signal lasts for a duration of 41.6μs (5.2μs X 8) before repeating. The INFO1 signal monitor is designed to indicate the presence of an INFO1 signal when six consecutive INFO1 signals have been recognised. The block diagram for the INFO1 monitor is shown in figure 3 of appendix B. All the hardware sub-blocks that are shown in figure 3 of appendix B operate synchronously with the same clock signal (MCLK_TENT). The timing diagram in figure 10 represents the system operation for the duration of six INFO1 signals. In the following signals are defined and the operation of each sub-block is described. The signals shown in the timing diagram are all generated within the INFO1 monitor macro.

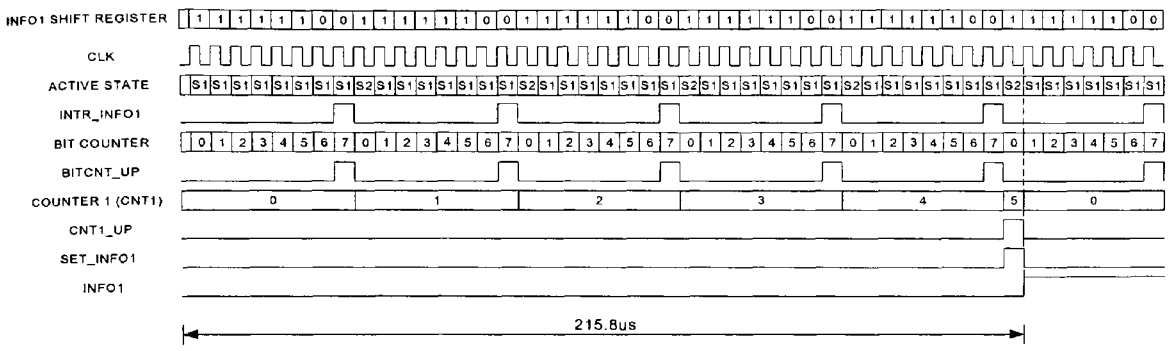


Figure 10 - The timing diagram for the INFO1 monitor.

The receiver clock S Bus Synchronisation and PLL circuit samples the S-Bus signal to produce a synchronised data stream (DATA_TENT). This synchronised data stream is applied to the INFO1 monitor. A serial shift-register within the INFO1 monitor is loaded with the synchronised data stream with the objective of identifying the repetitive INFO1 signal characterised by the binary sequence represented by 11111100. The shift-register signal in the timing diagram (INFO1 SHIFT REGISTER) indicates the binary data stream as it propagates through the shift-register. The dominant signal represented by the timing diagram is 11111100, indicating the presence of an INFO1 on the line. The INFO1 monitor hunts for this binary sequence and generates an interrupt signal (INTR_INFO1) when identified. The interrupt signal resets a bit counter whose state sequence (BIT COUNTER) is represented on the diagram. The bit counter realises a timer that produces a time-out signal (BITCNT_UP) when the INFO1 signal is complete. The time-out signal should coincide with the INFO1 interrupt signal each time a valid INFO1 signal has passed. A state machine controller monitors the consistency of the INFO1 signal by checking that the INFO1 interrupt is active when the time-out signal is active. The state machine is implemented by the INFO1_CON sub-block. The ASM chart for the state machine is shown in figure 11. As the state machine has three states, there are eight

possible states (2^3 = 8) that the state sequence could enter. The one-hot encoding method was employed during the state machine implementation process. The one-hot encoding method ensures that only one of the three state flip-flops is active at any one time. If more than one state flip-flop becomes active then the state machine will produce erroneous results. To avoid this a circuit is required to monitor the state sequence and reset the state machine if this situation occurs. The one-hot encoding method was employed for all state machines in the design. The state monitoring circuit was not implemented for the state machines in this design. It is recognised that this would be required for a commercial design where robust operation is required. The state sequence (ACTIVE STATE) is shown on the timing diagram. The objective of the state machine is to continually monitor the data from the ISDN line and record consecutive INFO1 signals. When a reset is active the default state (state S0) is entered. The state machine waits in state S1 for the signal BITCNT_UP to go active and then checks the state of INTR_INFO1 to determine the existence of an INFO1 signal. Two counters (MOD6_CNTR) are used by the process and these are designated counter 1 and counter 2 for the purposes of explanation. Both counters are decoded to produce an indication on reaching the sixth state. The indication signals are CNT1_UP for counter 1 and CNT2_UP for counter 2. Counter 1 maintains a record of the number of consecutive INFO1 signals that have been successfully detected. Counter 2 maintains a record of the number of consecutive situations where an INFO1 signal was not detected within the time-out. The state-machine advances counter 1 when an INFO1 signal is successfully identified. When six consecutive INFO1 signals have been detected counter 1 will produce an active CNT1_UP signal and the state machine will set the INFO1 flag to logic 1 with the signal SET_INFO1. If an absence of the INFO1 signal is detected over six time-out periods then the state

machine resets the INFO1 flag with the RST_INFO1 signal. The hardware sub-blocks will be described in the following.

The INFO1 shift-register (INFO1_REG).

The INFO1 shift-register (INFO1_REG) samples data from the ISDN telephone line and hunts for an INFO1 signal. When an INFO1 signal is identified an interrupt signal is generated (INTR_INFO1).

The INFO1 flag (INFO1_FLAG).

The INFO1 Flag (INFO1_FLAG) is a register, which is set to logic 1 to indicate the existence of an INFO1 signal. The input signal SET_INFO1 sets the flag to logic 1 and the signal RST_INFO1 resets the flag to logic 0. The signal SYNC indicates that synchronisation with the S bus has been established. When SYNC is active high the INFO1 flag is held low.

The INFO1 Bit Counter (MOD8_CNTR).

The INFO1 Bit (MOD8_CNTR) counter implements a three bit binary counter and thus has eight states. Each state lasts for 5.2μs and the counter repeats after 41.6μs. The bit counter is reset by the INFO1 interrupt signal and is used as a timer during the monitoring process. The input signal ADV enables the counter to operate. This is set permanently to logic 1. The input signal RST is a synchronous reset. The counter is reset when the INFO1 interrupt is active or when the RST_BITCNT signal from the state machine is active. The count sequence is decoded to produce the signal CNT_UP. CNT_UP which goes active when the binary count sequence is 111.

Counter 1 and Counter 2 (MOD6_CNTR).

Counter 1 and counter 2 are identical. The counter macro MOD6_CNTR is used in both cases. MOD6_CNTR implements a binary counter. The counter has six states and the count sequence is decoded to produce the signal MOD6_UP when the sixth state is active. The input signal ADV is a synchronous clock enable for the counter. The input signal RST is a synchronous reset.

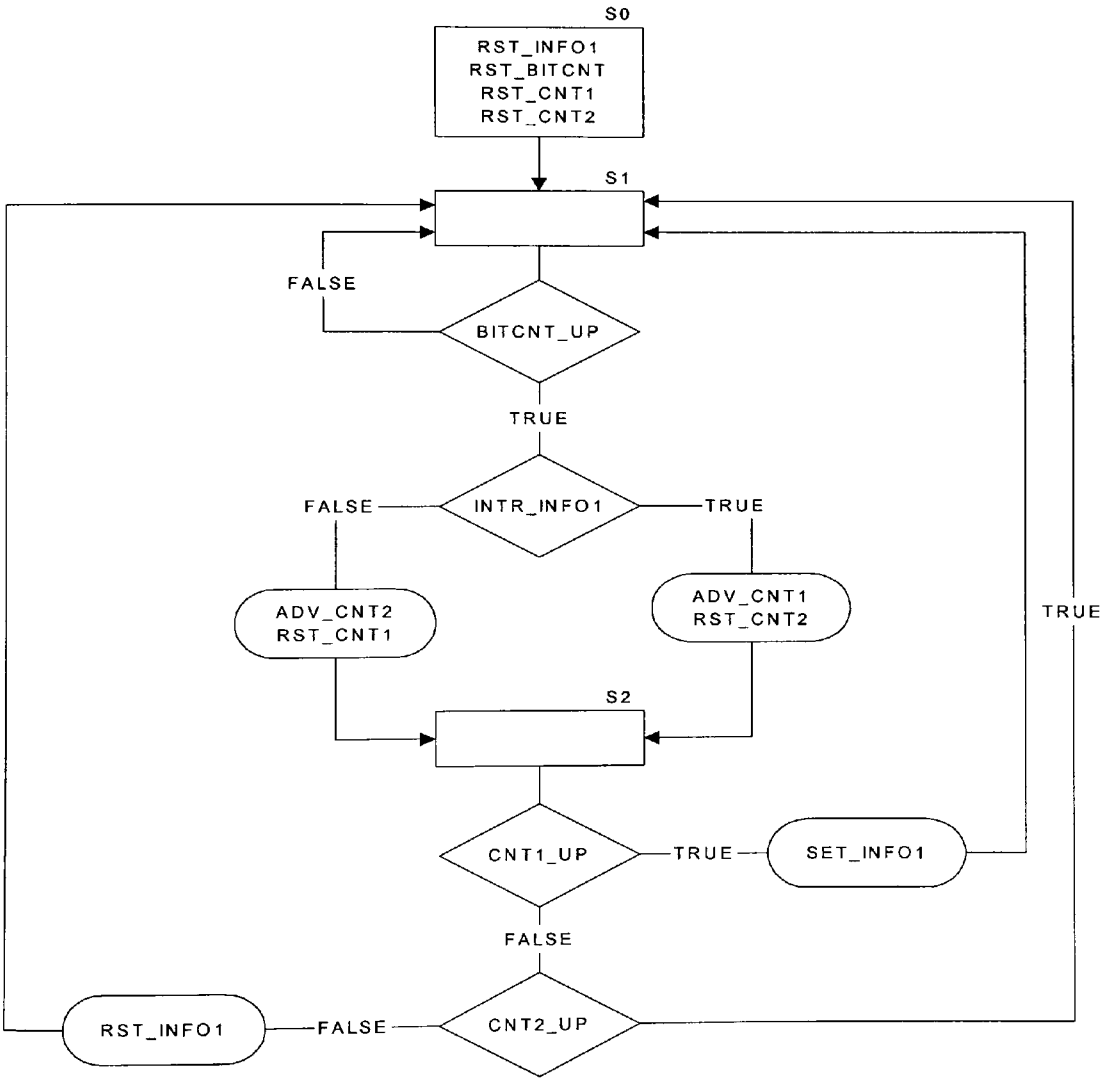


Figure 11 - The INFO1 monitor ASM chart.

3.1.3 The INFO2 signal monitor (INFO2_MON).

The INFO2 signal monitor hunts for the existence of an INFO2 signal on the S Bus. An INFO2 signal is transmitted from the NT to the TE. Recommendation I.430 defines an INFO2 signal as a frame with all bits of the B, D and E echo channels set to binary zero with the Activation bit (A bit) set to binary zero. The binary state of the A bit distinguishes the INFO2 signal from an INFO4 signal. Once synchronisation with the S bus is established, the INFO2 monitor continually reads the state of the A bit and maintains an up-to-date indication of signal status. The monitor is designed to indicate the presence of INFO2 when three consecutive frames of data with the A bit set to binary zero have been identified. A block diagram for the INFO2 monitor is comprised of four sub-blocks and is shown in figure 4 of appendix B. The timing diagram in figure 12 will first be presented to describe to overall operation of the system. The hardware components of the INFO2 monitor will then be explained.

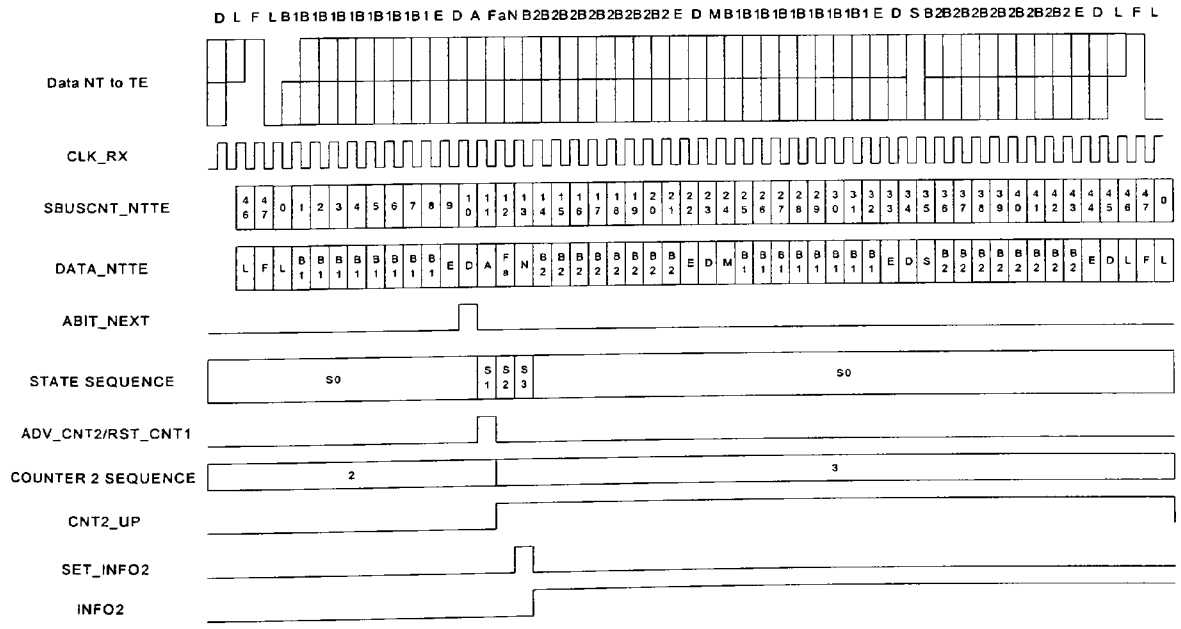


Figure 12 - The timing diagram for the INFO2 signal monitor.

The timing diagram shows one frame of data transmitted from the NT to a TE. The count sequence SBUSCNT_NTTE is produced by the Synchronisation and PLL circuit and identifies each bit within a frame of data. The A bit is defined by the count sequence as bit number eleven. The SBUSCNT_NTTE count sequence is decoded to produce the signal ABIT_NEXT. ABIT_NEXT goes active when the count sequence reads ten. A state machine (INFO2_CON sub-block – see Figure 4, appendix B) is employed to control the process and operates once every frame. The state machine is enabled once synchronisation with the S bus is established. Synchronisation is indicated when the SYNC signal is set high. The state sequence is shown on the timing diagram. The ASM chart for the state machine operation is shown in figure 14. The one-hot state machine encoding method was employed during the implementation stage. The state machine monitors the ABIT_NEXT signal and when ABIT_NEXT is active the state sequence moves from state S0 to state S1. State S1 lasts for one clock cycle. During state S1 the binary state of the A bit is established. The input signal PULSE is connected to the synchronised data stream sampled from the S Bus. If PULSE is logic zero an INFO2 signal is present. If PULSE is logic one an INFO4 signal is active. The state machine updates two counters (MOD4_CNTR) during state S1. The counters are designated counter one and counter two for explanation purposes. In the block diagram for the INFO2 monitor, U9 implements counter one and U10 implements counter 2. Each time an INFO2 signal is identified, counter two is advanced and counter one is reset. When a signal other than INFO2 is identified, counter two is reset and counter one is advanced. The counter one is checked during state S2 and if three consecutive non-INFO2 signals have been identified the counter output CNT1_UP will be active and the state machine will clear the INFO2 indication. During state S3 counter two is

checked and if three consecutive INFO2 signals have been identified the counter output CNT2_UP will be active. The timing diagram represents this case. When INFO2_UP is active the state machine sets the INFO2 signal indication. The time for the INFO2 monitor circuit to identify the presence of an INFO2 signal is shown by the timing diagram in figure 13 to be 585.8μs. The timing diagram represents four S Bus data frames. The data frames have been simplified to show the synchronisation pulses and the Activation symbol (A-bit) only. The Activation symbol may be a positive pulse or a negative pulse when encoded as a binary zero. The synchronisation pulses have already been explained in the previous sections. The Activation symbol is presented in the timing diagram so that the three possible signal states are emphasised i.e. a positive or negative pulse (Mark) or no signal level (space). The time taken for the INFO2 monitor to identify an INFO2 signal will be used as a verification check during the results analysis section in chapter 5. Once the INFO2 signal ceases to be transmitted the INFO2 detection circuit will cancel the INFO2 signal indication after three consecutive A-bits are identified as logic one. The total time to clear the INFO2 signal will be 250μs + 250μs + 72.8μs + 7.8μs = 580.6μs. The 13μs is reduced to 7.8μs in this case as the reset signal to clear the INFO2 indication occurs during state 2 of the ASM chart.

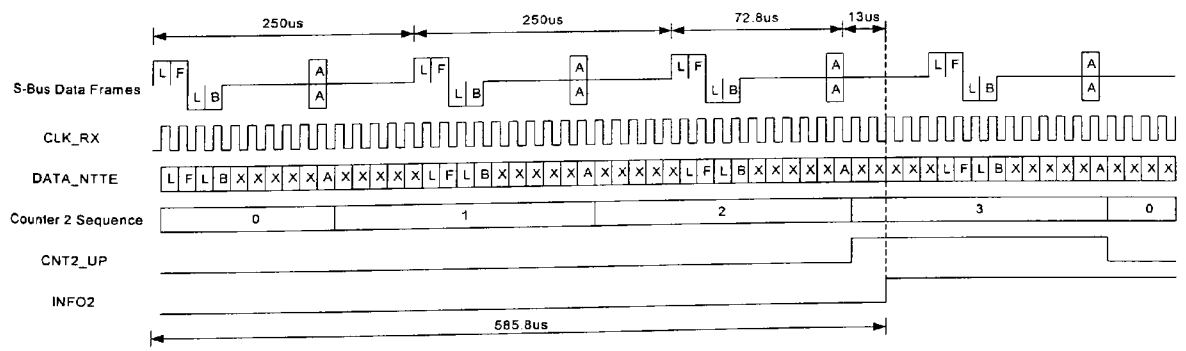


Figure 13 - The time required for INFO2 signal identification.

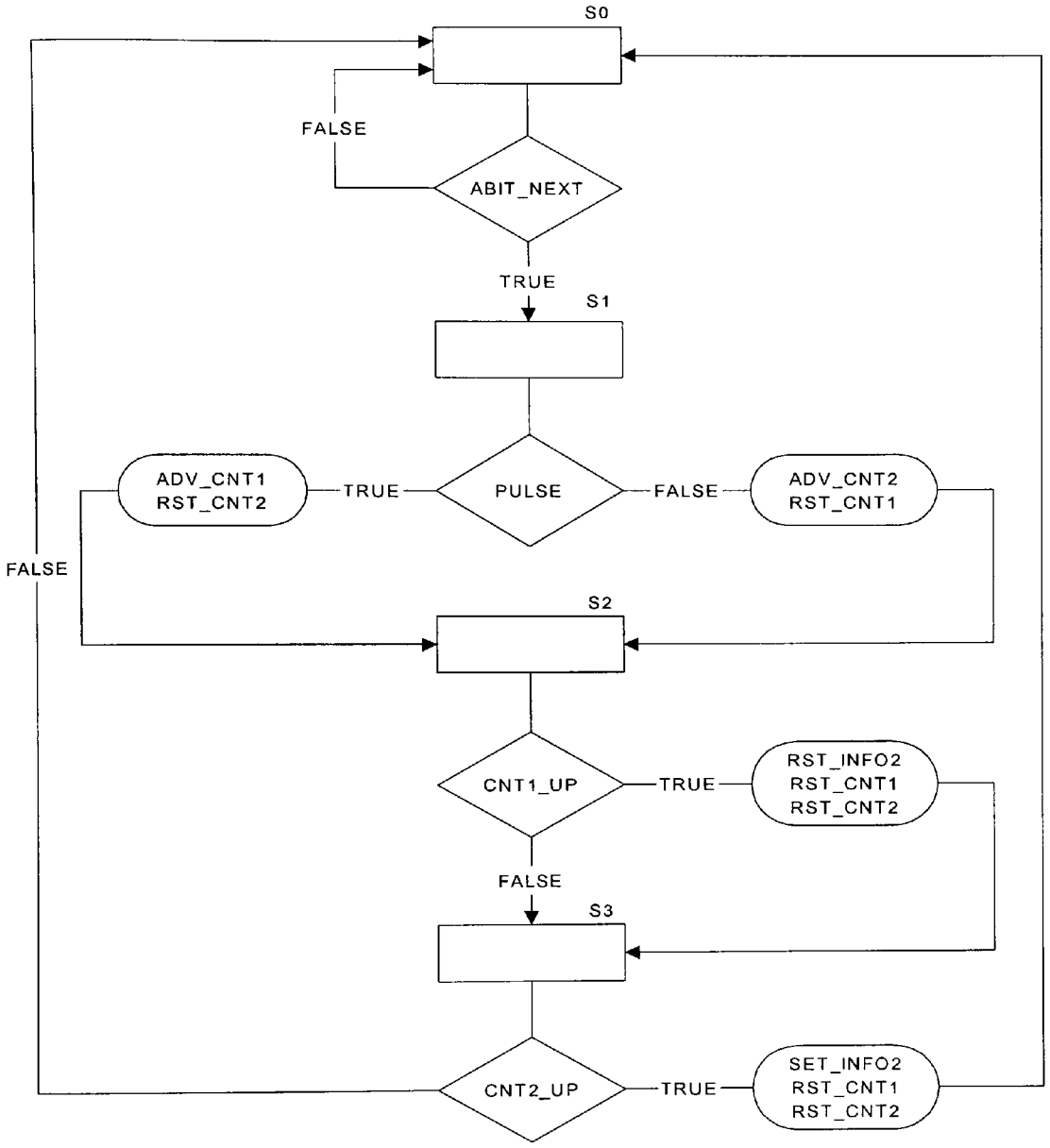


Figure 14 - The ASM chart for the INFO2 signal monitor.

The following is a description of the hardware components that facilitate the state machine during the INFO2 signal identification process.

The INFO2 Flag (INFO2_FLAG).

The INFO2_FLAG implements a D-type flip-flop. The signal SET_INFO2 from the state machine sets the INFO2 output signal high to indicate an active INFO2 signal. The signal RST_INFO2 from the state machine resets the INFO2 output signal low. All operations occur synchronously with the clock signal (CLK). When synchronisation with the S bus is lost the INFO2 signal is cleared.

Counter one and Counter two (MOD4_CNTR).

Counter one and two both use the MOD4_CNTR macro. The MOD4_CNTR implements a two bit binary counter. The counter state sequence is decoded to produce the output CNT_UP when the binary state sequence is 11.

3.1.4 The INFO0 signal monitor (INFO0_MON).

The INFO0 monitor samples the data on the S bus and indicates if an INFO0 is active on the line. An INFO0 signal is defined as a continuous signal composed of binary ones. If three consecutive frames of data are composed of binary ones then it can be assumed that an INFO0 is active. When a signal is being transmitted the minimum number of binary zeros that may exist within a frame is four. If at least four binary zeros exist in three consecutive frames then an INFO0 is inactive. This situation exists when the two coding rule violations are the only symbols being transmitted as logic zeros within a frame.

The block diagram for the INFO0 signal monitor is shown in figure 5 of appendix B. The design is composed of five sub-blocks. The timing diagram in figure 15 details the INFO0 identification process. The signal on the S Bus is labelled as S-Bus signal on the timing diagram. The input signals to the INFO0 monitor are the sampled S bus binary data stream (DATA), a time–out signal from the S bus PLL which goes active at the end of each data frame (NEXT_FRAME) i.e. every 250μs, and the clock signal (MCLK). A global hardware reset is used. Therefore an extra reset signal is not required. The clock signal (MCLK) is the S bus data-sampling clock derived from the S bus PLL processing. The monitor employs four binary counters. The following explains the operation of each counter and reference will be made to the block diagram for the INFO1 monitor throughout. U10 (INFO0_CNT2) on the diagram will be designated counter 2. U9 (INFO0_CNT1) on the diagram will be designated counter 1. U11 (INFO0_CCNT2) on the diagram will be designated counter 3 and U6 will be designated counter 4. Counter 1 and Counter 2 are synchronously reset when the signal, NEXT_FRAME goes active high. The sampled binary data stream from the S bus enables Counter 2 when a binary one is transmitted. Counter 2 implements a binary counter and its count sequence is decoded to produce CNT2_UP. Counter-2 has 48 states and are numbered here as 0 to 47. The timing diagram in figure 15 shows the count sequence for Counter-2. The timing diagram is simplified to show part of the count sequence. An N in this case indicates the next count in the sequence. CNT2_UP goes active for one clock cycle when the count sequence reads 47. Should the sampled data stream consist entirely of binary ones for one frame, Counter 2 would advance through each state until state number 47 is reached when the CNT2_UP signal becomes active. CNT2_UP would go active every 250μs should the binary data stream consist of binary ones. The CNT2_UP signal

becomes the clock enable signal for Counter 3. Counter-3 is a binary counter with four states and is used to record the number of frames in which an INFO0 is transmitted. When Counter-3 is advanced to its fourth state the signal SET_INFO0 is set high for one clock cycle. This sets the INFO0 signal flag to logic one indicating the presence of an INFO0 signal. The time period required for identification of an INFO0 signal is shown by the timing diagram to be 760μs.

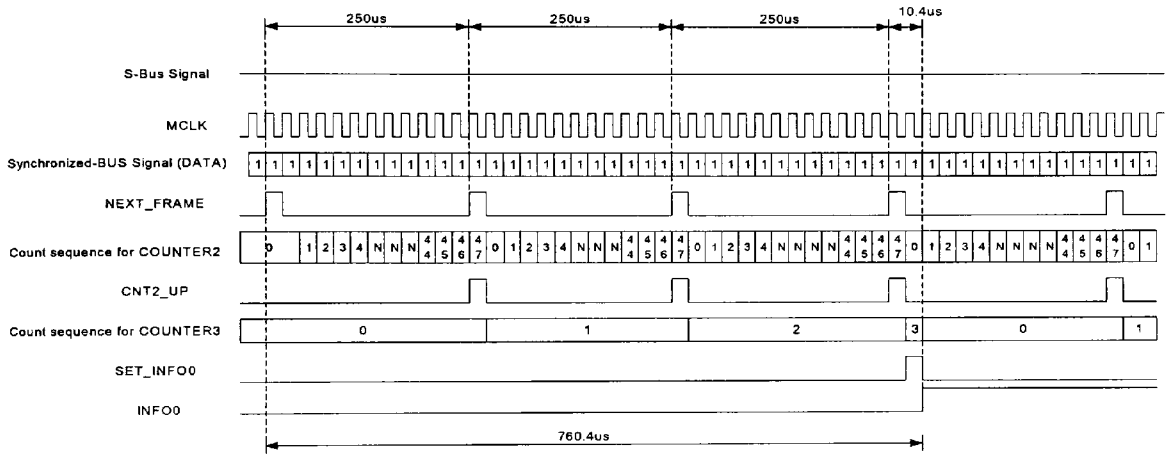


Figure 15 - The time period required for an active INFO0 identification.

The timing diagram shown in figure 16 details the signals associated with identification of an inactive INFO0 signal. The following explains the process. The sampled binary data stream is inverted and connected to the clock enable input of Counter 1. Counter 1 is advanced when a logic zero exists in the data stream. Counter 1 implements a binary counter with four states. If three or more binary zeros exist on the line, then Counter 1 will be advanced and the signal CNT1_UP will go active for one clock cycle. The signal CNT1_UP from Counter 1 provides the clock enable

signal for Counter 4. Counter 4 works in the same way as Counter 3. When three non-INFO0 frames have passed the signal CLR_INFO0 resets the INFO0 flag. The timing diagram represents the worst-case condition when only four pulses are transmitted within a 250μs period. The time period required to clear an INFO0 condition under the circumstances presented in figure 16 is 768.2μs. If more pulses are transmitted within 250μs then the INFO0 indication will be cleared sooner. Three consecutive instances of either an INFO0 or a non-INFO0 signal must be identified before the INFO0 monitor decides to indicate a chance in condition. This is achieved by ensuring that the identification of one signal condition causes a reset to the circuitry recording the existence of the other signal condition. For example if Counter 2 identifies an INFO0 signal condition, the resulting signal CNT2_UP will reset Counter 4 while advancing Counter 3.

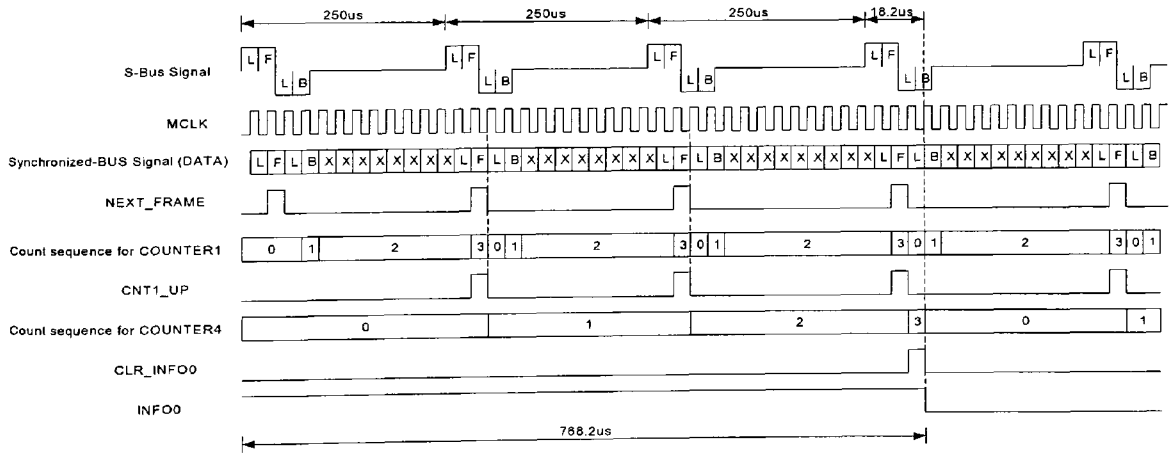


Figure 16 - The time period required for inactive INFO0 identification.

3.1.5 The INFO signal decoder.

Two INFO signal decoders are employed. The signal state on the ISDN for both directions of data transmission is monitored. The TE to NT signal decoder

(TENT_INFO_CON) monitors the signal state in the TE to NT direction of data transmission. The TE to NT signal decoder monitors the output signals from the INFO1 signal monitor, INFO0 signal monitor and the synchronisation indication from the S Bus PLL. One of the output signals TENT_INFO0, TENT_INFO1, TENT_INFO3 will be active high to indicate the currently active signal on the line.
A logic one produced on the signal TENT_INFO0 indicates an INFO0 signal in the TE to NT direction of data transmission. The same naming convention logic is applied to the other signals. The NT to TE signal decoder (NTTE_INFO_CON) provides an indication of the active signal in the NT to TE direction of data transmission. The output signals NTTE_INFO0, NTTE_INFO2, and NTTE_INFO4 indicate the currently active signal. In both cases only one signal indication will be active at any time. The absence of an active signal indicates that the signal on the S Bus is undefined.

3.2 The IOM-2 Bus Clock Generator (FSC_GEN block).

The IOM-2 Bus clock generator generates a frame synchronisation clock (FSC) and a data clock (DCL) to facilitate data transmission on the IOM-2 bus toward the Motorola 68302 micro controller. An IOM-2 data frame lasts for 125μs. The nominal frequency of the FSC clock is 125μs (8kHz). The nominal frequency of the DCL clock is 1.536MHz (650ns). There are two IOM-2 data frames for every one ISDN S-Bus frame. The timing diagram showing the relationship between the S bus and IOM bus can be seen in figure 17. The timing diagram is not to actual scale. The FSC clock is a frame synchronisation signal and provides an indication of the beginning of an IOM-2 frame. The DCL clock is the data clock and is used to synchronise data

transfer to the Microcontroller. The FSC clock generation circuitry is clocked with the 15.36MHz master clock. The block diagram for the IOM-2 bus clock generator is shown in figure 6 of appendix B. The generator is composed of the FSC clock generator circuit (FSC_ARCH) and the DCL clock generator circuit (DCL_ARCH). The six-bit bus labelled as SBUS_NTTE represents the S Bus count sequence. The FSC clock generator derives the FSC clock from the S bus count sequence. FSC is set high when the count sequence reads zero or twenty-four. The FSC clock is set low when the count sequence reads eight or thirty-two. The FSC clock generation process is enabled when synchronisation is achieved with the S bus. The FSC clock generator monitors the SYNC signal and resets the FSC generator on identifying a positive going edge.

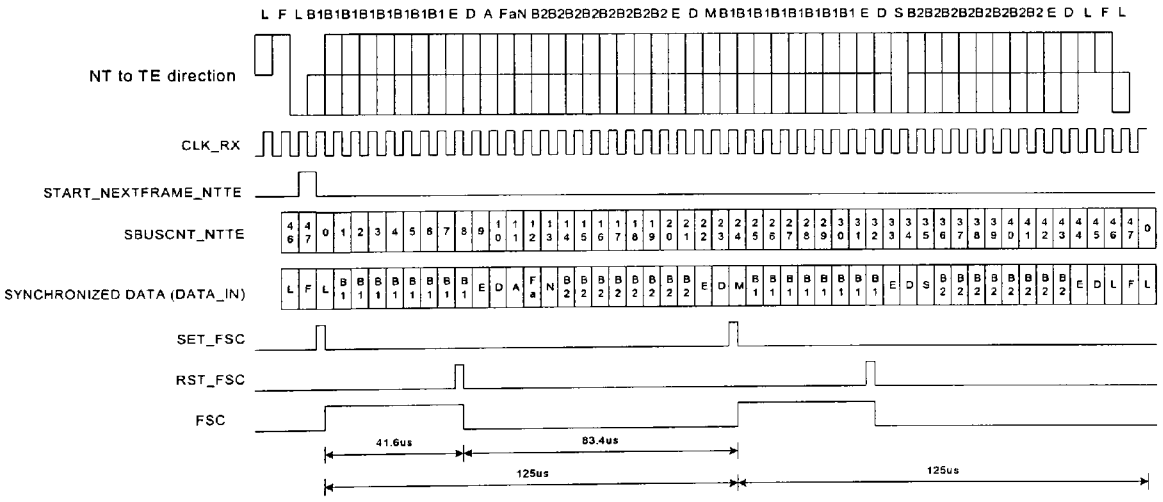


Figure 17 - Timing diagram for the FSC clock generation.

The 15.36MHz master clock is used to clock the DCL clock generator. The DCL clock generator is reset when a positive going edge is identified on the FSC clock signal. The generator uses a binary counter with ten states to derive the signals that set and reset the DCL clock signal. The timing diagram for the process is shown in figure

18. The timing diagram shows how the positive going edge of FSC is used to produce the reset signal (FSC_EDGE) synchronously with the 15.36MHz clock. The signal FSC_EDGE resets a binary count sequence. The binary count sequence can be expressed in integer form ranging from 0 to 9. This is expressed on the timing diagram with the signal COUNT. Each count state lasts for 65ns. The DCL clock signal is set high when the count sequence reads 9 and set low when state 4 is active. The setting and resetting of the DCL clock occurs synchronously with the 15.36MHz clock. This produces a 50% duty cycle with a clock repetition period of 650ns (65ns X 10).

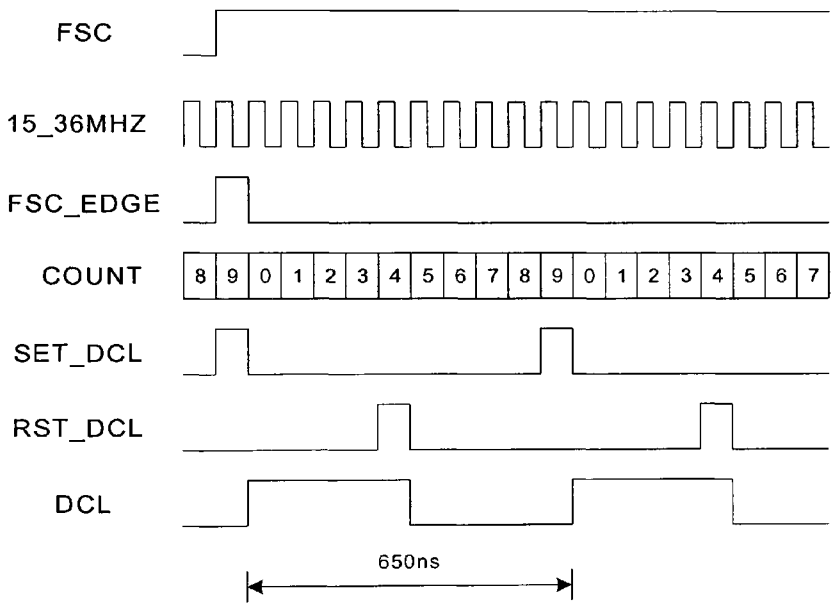


Figure 18 - Timing diagram for the DCL clock generation.

3.3 The IOM-2 Bus Interface (The IOM_BUS block).

The IOM-2 Bus Interface transmits the D and E channel information on the IOM-2 data up line (IOM_DU) towards the Microcontroller for analysis. The IOM-2 Bus interface also encodes the INFO signal status into the Command/Indication channel

codes shown in table 1. The DCL clock signal is used to clock the logic within IOM-2 Bus interface. Synchronous design techniques have been adopted throughout. The implementation focuses on supporting the IOM-2 channel number zero only. The FSC clock signal envelops the IOM-2 channel number zero when set to logic one. There are sixty-four DCL clock cycles within the time that the IOM-2 channel zero is active. There are two DCL clock periods for every data bit period in an IOM-2 frame. The block diagram for the IOM-2 Bus Interface circuit is shown in figure 7 of appendix B. The IOM-2 bus interface is composed of five sub-blocks. Each block will be explained in the following section. During the following sections reference will be made to the timing diagram for the IOM-2 bus interface as shown in figure 19.

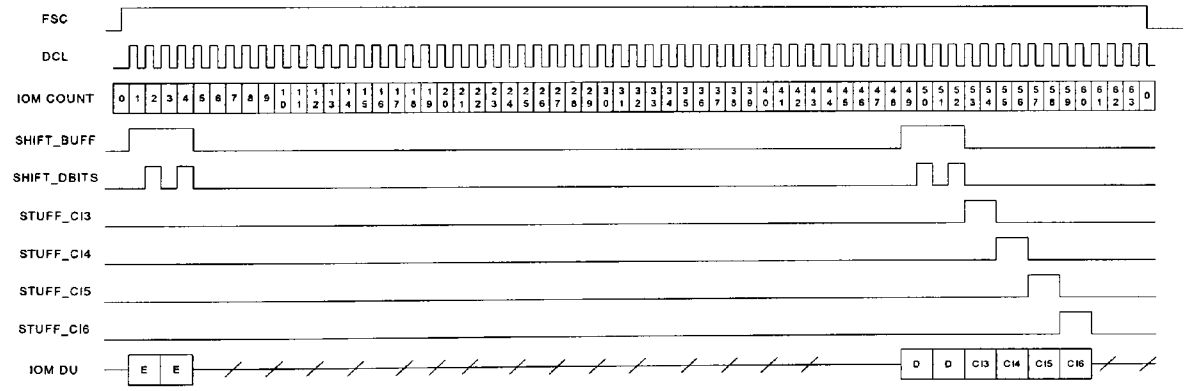


Figure 19 - Timing diagram for the IOM-2 Bus interface.

The IOM-2 shift-register.

The IOM-2 shift-register (DBITS_LOAD_ARCH) implements a parallel in to serial out shift-register and is clocked with the DCL signal clock. The E and D channel data extraction from the ISDN S Bus produces the data format shown in figure 20. The

data format represents the contents of the D channel shift-register located in the S Bus Interface block. As data is extracted from the S Bus, it is shifted serially from left to right along the shift-register storage locations. Eight bits are extracted from the S Bus. The first E bit extracted will be labelled Ea, the second Eb and so on. The same applies to the extracted D bits. When the S Bus Interface circuit has processed a frame of S Bus data, its D channel shift-register holds the parallel data (D_BITS). D_BIT(7) will hold Dd and D_BIT(0) will hold Ea. The input signal LATCH_DBITS synchronously latches the D channel data extracted from the ISDN S Bus into the IOM-2 shift-register within the IOM-2 Interface block. When data is latched into the IOM-2 Shift-register the data format is rearranged such that the format shown in figure 21 is achieved. The IOM-2 interface subsequently shifts the data out serially into the IOM-2 Multiplexer circuit. The rearranged D and E channel data is routed through the IOM-2 Multiplexer and onto the IOM-2 data-up line toward the Microcontroller.

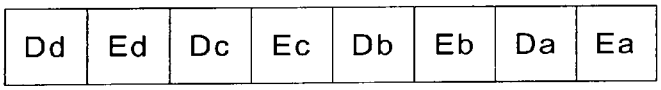


Figure 20 - Extracted D and E channel data format.

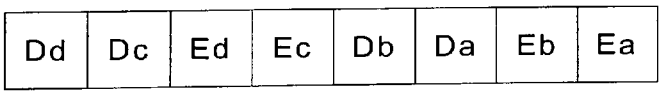


Figure 21 - Rearranged D and E channel data format.

The IOM-2 frame bit counter (IOM_CNT_ARCH).

The information to be transmitted toward the Microcontroller is inserted into the IOM-2 data-up line (IOM_DU) at the appropriate times defined by the IOM-2 frame bit count sequence (IOM COUNT). The relationship between the IOM-2 frame bit

count sequence and the data on the IOM-2 bus is shown on the timing diagram of figure 19. The count sequence ranging from 0 to 63 is derived from a binary counter clocked with the DCL clock. Two E bits and two D bits are transmitted per IOM-2 frame. The E bits are transmitted in the first two bits of the B1 channel of the IOM-2 bus. The IOM-2 count sequence is decoded to produce a data transmit enable signal (SHIFT) when the IOM-2 count sequence reads 1, 2, 3, 4, 49, 50, 51 or 52. This signal enables the IOM-2 Multiplexer and data is routed through to the IOM-2 data-up line. The E bit data is shifted serially into the IOM-2 Multiplexer input line (IOM_DCHAN_DATA) when the count sequence reads 1, 2, 3 or 4. The D bits are shifted serially into the IOM-2 Multiplexer input when the count sequence reads 49, 50, 51 or 52. Once a D or E channel bit has been transmitted the IOM-2 shift-register is enabled with the SHIFT_BUFF signal and the next bit of information is shifted out and transmitted next. The IOM-2 Shift-register is enabled when the count sequence reads 2, 4, 50 and 52.

The Command/Indicate encoder channel.

The currently active signal status is reported to the Microcontroller via the Command/Indicate channel. The Command/Indicate channel is a four-bit channel. The Command/Indicate channel is active when the IOM-2 frame bits count sequence reads 53 through 60. The individual bits within the channel are labelled as CI3, CI4, CI5 and CI6. The CI3 and CI4 encoder (CI3_CI4_ENCODE_ARCH block) encodes the INFO signal status for the NT to TE direction of data transmission. The CI3 and CI4 signals are encoded according to table 1. The CI5 and CI6 encoder (CI5_CI6_ENCODE_ARCH block) encodes the INFO signal status for the NT to TE

direction of data transmission. The encoding scheme for CI5 and CI6 signals can also be seen in table 1. In table 1 an X indicates an undefined signal.

ACTIVE INFO SIGNAL	NT to TE direction		TE to NT direction	
	CI3	CI4	CI5	CI6
UNDEFINED	0	0	0	0
INFO0	0	1	0	1
INFO1	x	x	1	0
INFO2	1	0	x	x
INFO3	x	x	1	1
INFO4	1	1	x	x

Table 1 - The Command/Indicate Encoder table.

The IOM Multiplexer.

The IOM Multiplexer (IOM_MUX) passes either D or E channel data or Command/Indication channel status information to the IOM data up line. The data up line is normally held in tri-state when not transmitting data. The multiplexer output directly drives a tri-state buffer as shown in figure 22. When a zero is on the multiplexer output, the tri-state buffer is enabled and the data up line (DU_PAD) is pulled low. Otherwise the tri-state buffer remains in high-impedance mode.

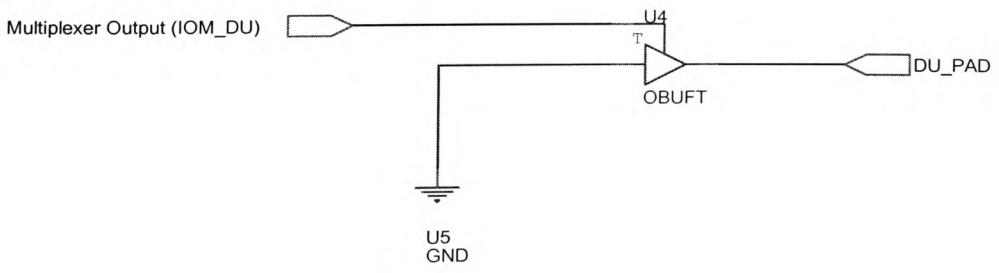


Figure 22 - The Tri-state buffer.

The select inputs to the multiplexer are derived from the IOM-2 count sequence. The select inputs STUFF_CI3, STUFF_CI4, STUFF_CI5, STUFF_CI6 and SHIFT become active at the times specified in the timing diagram of figure 19. The following VHDL code is presented here as it best describes the multiplexer operation.

```
library IEEE;
use IEEE.std_logic_1164.all;

entity IOM_MUX is
port (
      DBITS: in STD_LOGIC;
      CI_3: in STD_LOGIC;
      CI_4: in STD_LOGIC;
      CI_5: in STD_LOGIC;
      CI_6: in STD_LOGIC;
      STUFF_CI3: in STD_LOGIC;
      STUFF_CI4: in STD_LOGIC;
      STUFF_CI5: in STD_LOGIC;
      STUFF_CI6: in STD_LOGIC;
      SHIFT: in STD_LOGIC;
      IOM_DU: out STD_LOGIC
     );
end IOM_MUX;


architecture IOM_MUX_arch of IOM_MUX is

signal EN_3STATE: STD_LOGIC;
signal D_OUT: STD_LOGIC;
signal CI3_OUT: STD_LOGIC;
signal CI4_OUT: STD_LOGIC;
signal CI5_OUT: STD_LOGIC;
signal CI6_OUT: STD_LOGIC;


begin


EN_3STATE <= not ((STUFF_CI3) and (STUFF_CI4) and (STUFF_CI5) and (STUFF_CI6) and
(SHIFT));

D_OUT <= DBITS and SHIFT;

CI3_OUT <= CI_3 and STUFF_CI3;
CI4_OUT <= CI_4 and STUFF_CI4;
CI5_OUT <= CI_5 and STUFF_CI5;
CI6_OUT <= CI_6 and STUFF_CI6;

IOM_DU <= EN_3STATE or (D_OUT or CI3_OUT or CI4_OUT or CI5_OUT or CI6_OUT);


end IOM_MUX_arch;
```

Implementation results of the FPGA place and route process.

The Firmware design was targeted at a Xilinx 4010 FPGA. The place and route results below summarise the device utilisation. The following is copied from the place and route file produced by the Xilinx design tools.

Device utilisation summary:

Number of External IOBs 8 out of 61 13%
Flops: 0
Latches: 0
Number of Global Buffer IOBs 1 out of 8 12%
Flops: 0
Latches: 0

Number of CLBs 292 out of 400 73%
Total Latches: 0 out of 800 0%
Total CLB Flops: 208 out of 800 26%
4 input LUTs: 530 out of 800 66%
3 input LUTs: 52 out of 400 13%

Number of BUFGLSs: 4 out of 8 50%
Number of STARTUPs: 1 out of 1 100%

Chapter 4

4 Test procedure for the FPGA design.

This chapter focuses on the digital simulation of the firmware design. In this chapter the VHDL code representing the firmware design will be referred to as the device under test (DUT). The operation of the DUT was verified using a VHDL test bench. The VHDL Test bench was written to apply test vectors to the DUT and monitor the resulting signals. The test bench can be partitioned into a number of components. Each component performs a specific task within the test procedure. A block diagram for the VHDL test bench is shown in figure 23. The objective of the test is to simulate the signals flowing between an NT and TE during an ISDN call set-up procedure.

The NT to TE Data Generator produces the signals flowing in the NT to TE direction of data transmission. The TE to NT Data Generator services the TE to NT direction of data transmission. D and E channel test vector information is read from a text file and inserted into the ISDN S Bus data frames flowing between the NT and TE. Two text files containing D channel information (D_BITS.TXT) and E channel information (E_BITS.TXT) are required. The objective of the test is to initiate data transmission between the TE and NT. The DUT should extract the D and E bits from the S bus and transmit them on the IOM-2 bus. The resulting D and E channel information is read from the IOM-2 bus and written to individual text files (EBITS_RESULT.TXT and DBITS_RESULT.TXT) by the IOM-2 Monitor.

The test bench generates the 15.36MHz clock (C15_36MHZ) and is used to derive the master clock (MCLK_SIM) and the timing events produced by the Synchronisation signals generator. The Test sequence is the master controller for the test procedure. The Test sequence generator turns the INFO signals associated with

the simulated S Bus on and off. The Test sequence generator monitors the INFO signal indications produced by the DUT. The state of the INFO signals is written to a text file (INFO_SIG_RESULT.TXT) every time there is a change in signal status. In the following section each of the Test bench components are described.

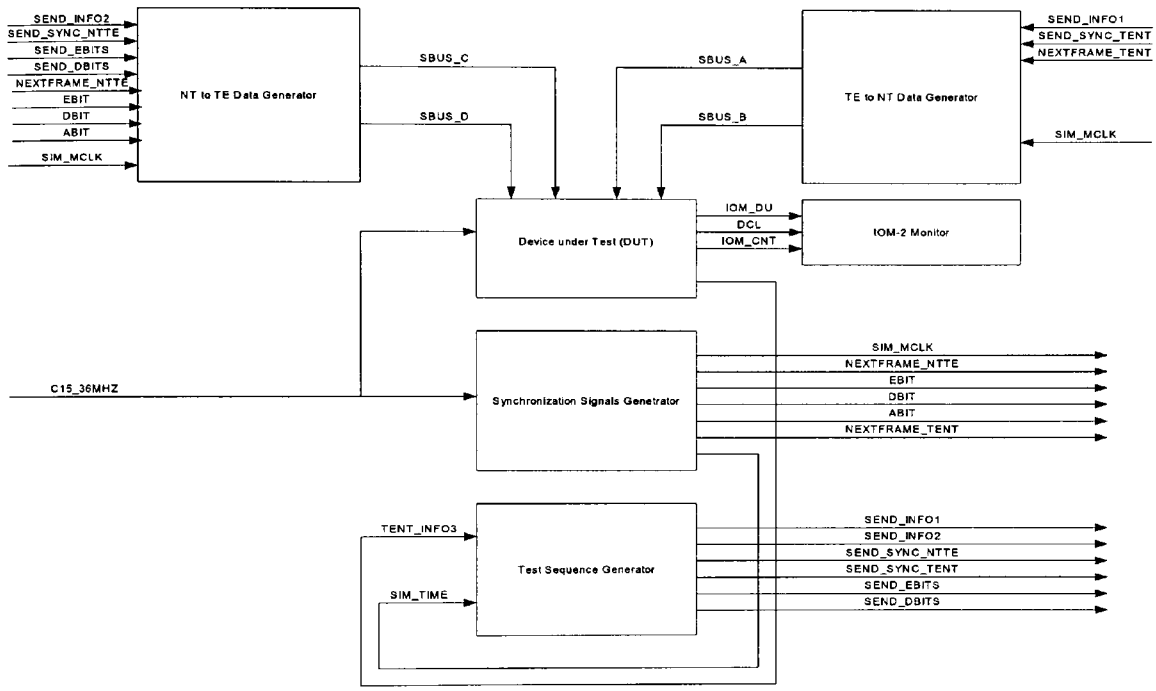


Figure 23 - The VHDL Test bench block diagram.

4.1 The Test Sequence Generator.

The timing diagram for the Test Sequence Generator is shown in figure 24. The timing diagram represents the simulated synchronisation signal sequence over an 8ms time period. The Test Signal Generator controls the operation of the test sequence. The signals for both directions of transmission are shown. The test sequence simulates

a normal ISDN physical layer connection set-up between an NT and a TE. The sequence starts with the NT and TE inactive, and transmitting an INFO0 signal. This situation lasts for 1mS. The INFO1 signal transmission is enabled between the TE and the NT when the SEND_INFO1 signal is set to logic 1. The INFO2 signal transmission is enabled when the signals SEND_SYNC_NTTE and SEND_INFO2 are both set to logic one. The signal SEND_SYNC_TENT is set high by the Test Signal Generator at 4ms to enable the transmission of an INFO3 between the TE and NT.

The DUT is monitored for an indication that INFO3 has been successfully identified. It is expected that the circuit under test should correctly identify the existence of an INFO3 signal within three frames. This indication should occur roughly at 4.75ms and has been indicated with a hashed line on the timing diagram. At this point in the simulation both INFO3 and INFO4 signals are active and the insertion of D and E bits onto the S Bus may begin. The signals SEND_EBITS and SEND_DBITS enable the transmission of D and E channel information. The DUT should echo this information in the IOM-2 bus if the circuit is operating correctly.

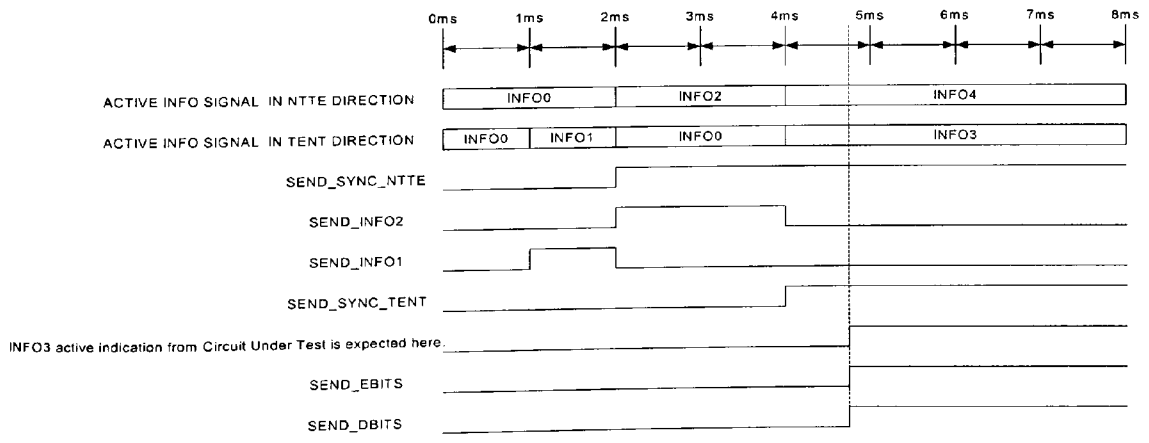


Figure 24 - The timing diagram for the Test Sequence Generator.

The Test Sequence Generator is written in behavioral VHDL style. The software flow chart for the Test Sequence Generator is shown in figure 25a and 25b. The software loops through the sequence shown enabling the transmission of the INFO signals in accordance with the timing diagram already discussed. The input variable SIMTIME holds the running simulation time. This is read continuously and when the simulation time has advanced past 8ms the simulator is stopped. The software procedure UPDATE_INFOSIG_RESULT writes the status of the INFO signals in both directions of transmission to a text file (INFO_SIG_RESULT.TXT). Once the test sequence is complete the text file may be analysed manually.

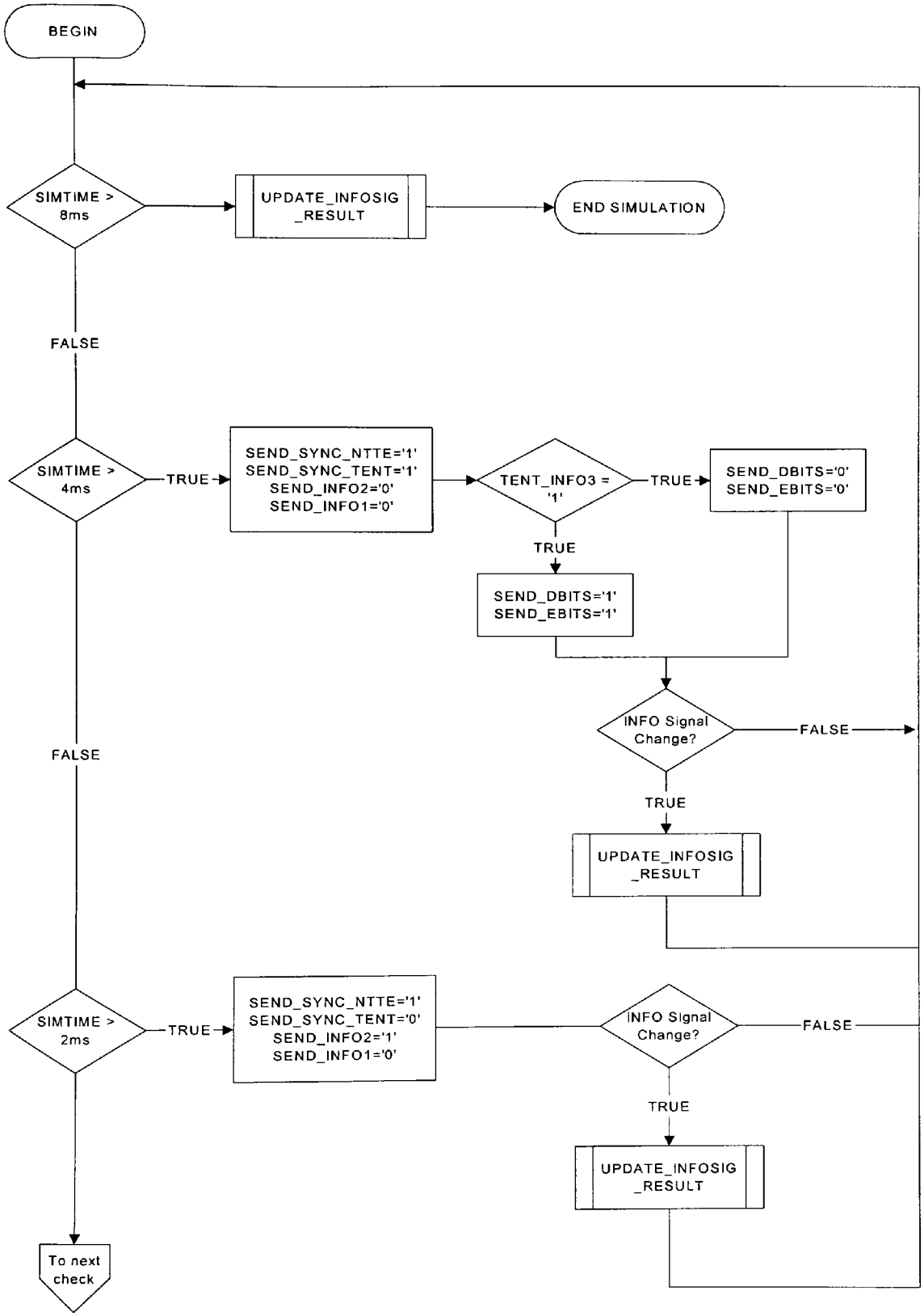


Figure 25a - The software flow chart for the Test sequence generator.

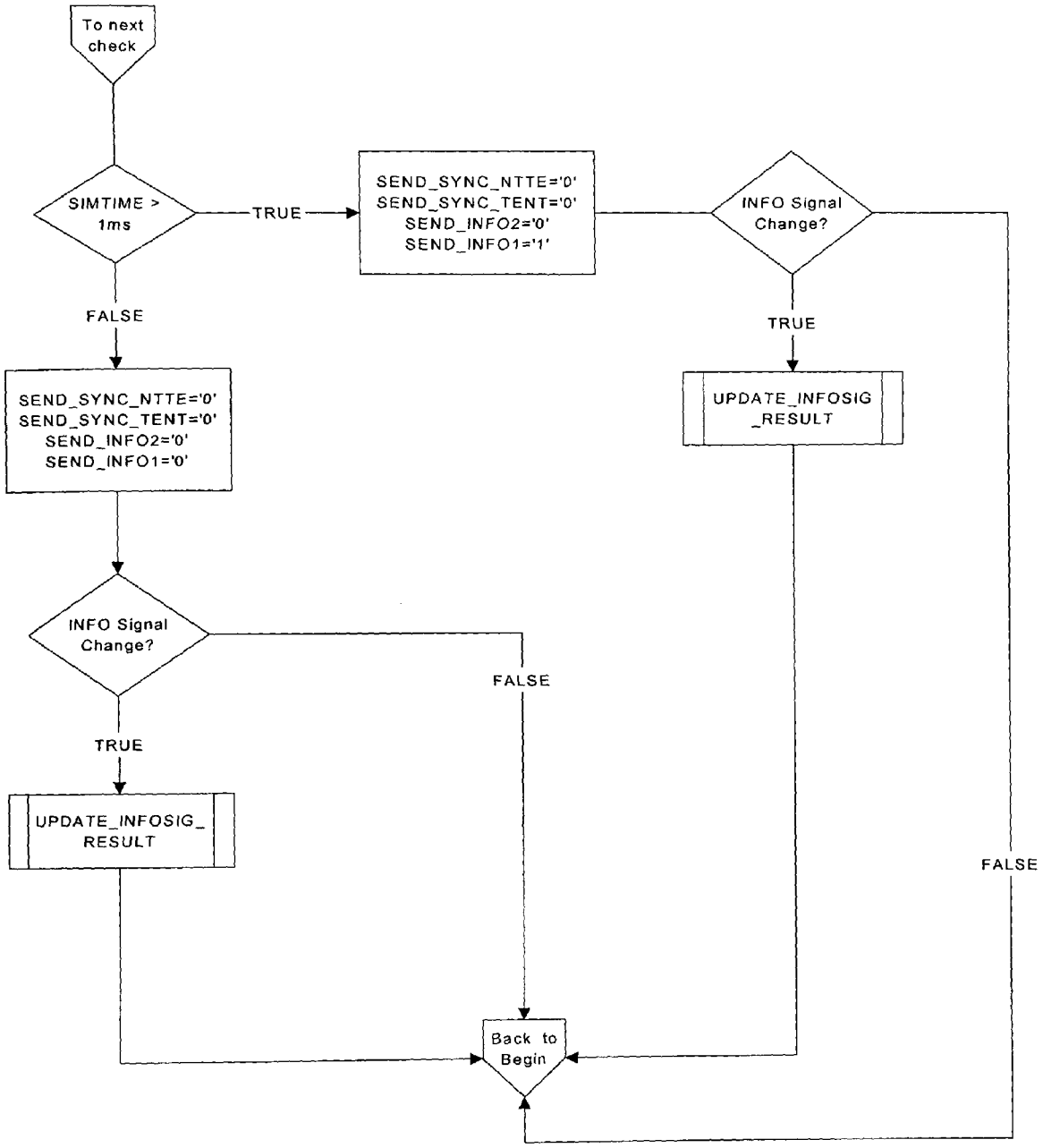


Figure 25b - The software flow chart for the Test Sequence Generator.

4.2 Synchronisation Signals Generator.

The Synchronisation signals generator has two tasks to perform. Task one is the generation of the master clock and the synchronisation signals for the Test bench. The nominal bit rate on the S Bus, as stated in Recommendation I.430 is 192 kilo-bits-per second (kbps). The master clock (SIM_MCLK) operates at a frequency of 192kHz. This is used to provide the clock signal for the NT to TE Data Generator block and the TE to NT signal Generator block. The timing diagram presented in figure 26 shows the master clock generation process. The Test bench's 15.36MHz clock signal is used to clock a count sequence that runs from 0 to 39 (TICKCOUNT). The count sequence is monitored to produce the signal TOGGLE_SIM_MCLK each time the count sequence reads 39. The timing diagram is not to scale. The letter N represents the count sequence between 1 and 38. This reduces the detail by compressing the detail. The signal TOGGLE_SIM_MCLK is then used to set and reset the signal SIM_MCLK.

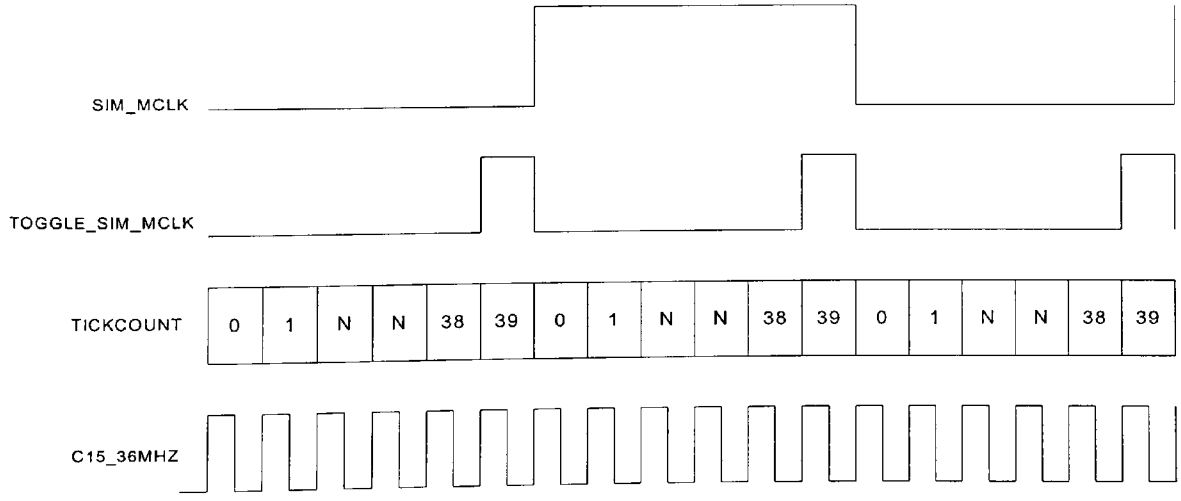


Figure 26 - The timing diagram for the master clock generation.

The second task is to generate the signals required to synchronise the different test bench components. The simulation master clock is used to clock a count sequence that defines the simulated S Bus. The count sequence runs from 0 to 47 and is advanced every 5.2μs. The S Bus can be derived directly from the count sequence. For the purpose of this test procedure a convention has been established that links each state within the count sequence to an individual S Bus symbol. The timing diagram shown in figure 27 demonstrates how the simulated count sequence SBUSCOUNT relates to the simulated S BUS. The S Bus data frames flowing in the NT to TE direction are offset by two bit periods with respect to the data frames flowing in the TE to NT direction. The signal NEXTFRAME_NTTE occurs once every 250μs when the count sequence reads 47. This signal is used to synchronise the generation of data frames for the NT to TE direction of data transmission. The signal NEXTFRAME_TENT is used to synchronise the generation of data frames for the TE to NT direction of data transmission. NEXTFRAME_TENT is decoded from the count sequence such that there is an offset of two count states from the signal NEXTFRAME_NTTE.

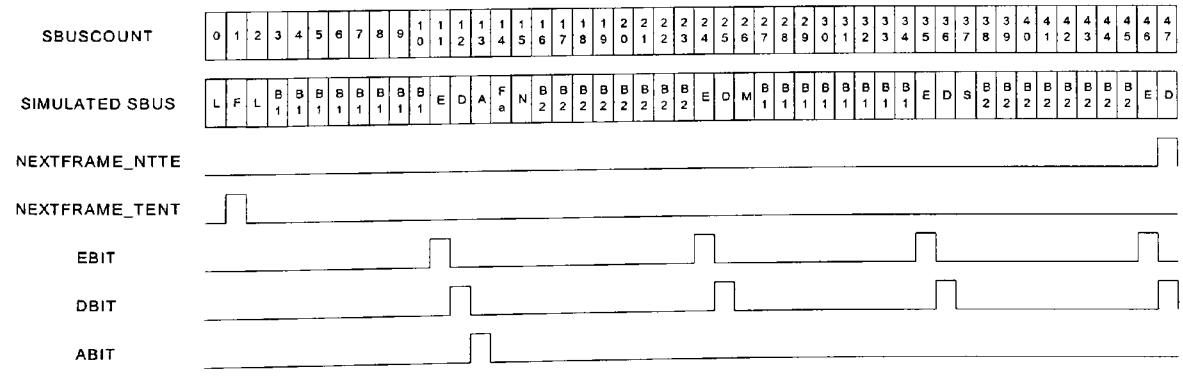


Figure 27 - The timing diagram for the synchronisation signals generation process.

4.3 The NT to TE Data Generator.

The NT to TE Data Generator produces one S Bus data frame every 250μs. The objective is to simulate the signals produced by the analogue interface circuit when connected to the S Bus. The generator is implemented as a state machine. The timing diagram for the process is shown in figure 28. The signals SBUSRX_C and SBUSRX_D represent the positive and negative pulses on the S Bus in the NT to TE direction. The NT to TE Generator simulates the S Bus synchronisation pulses by producing the coding rule violations as specified in ITU-T Recommendation I.430. The count sequence generated by the Synchronisation Signals Generator is shown again here in order to demonstrate the relationship between the simulated S Bus and the signals associated with the S Bus signal generation process.

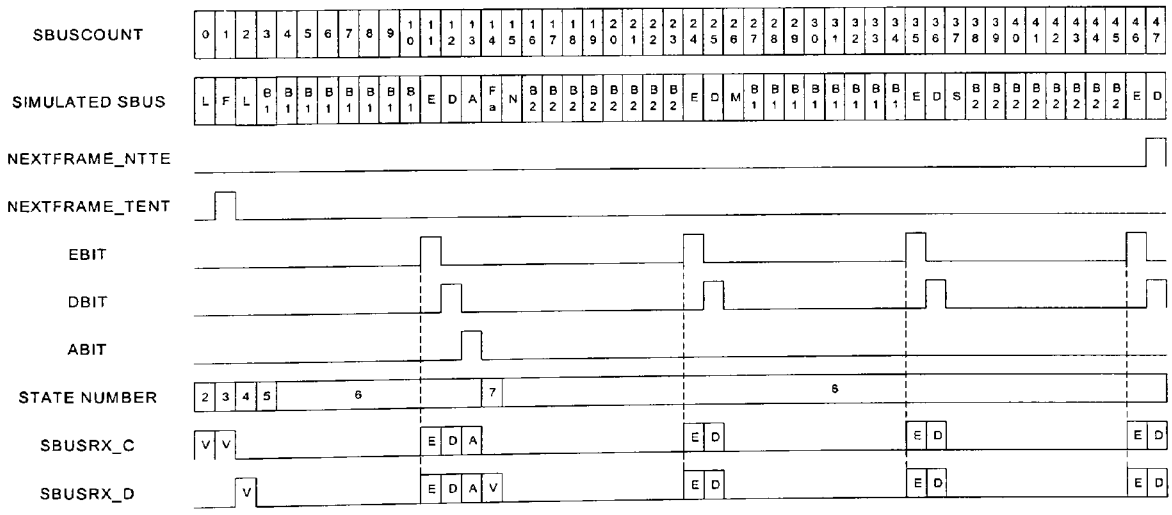


Figure 28 - The timing diagram for the NT to TE Data Generator State Machine.

The state machine has eight states. . The state machine operation is presented as an algorithmic state machine (ASM) chart in figure 29a, 29b and 29c. The one-hot state machine encoding method was used during the implementation process. Each state is allocated a state number as indicated in the timing diagram. State number 0 and 1 are wait states and are not shown in the timing diagram. The reset signal for the test bench forces the state machine into state number 0. The Test Sequence Generator enables the NT to TE Data Generator when the signal SEND_SYNC_NTTE is high. A pulse generated on either of the signals SBUSRX_C and SBUSRX_D represents a binary zero. Pulses are labelled on the timing diagram to represent their function within the data frame. Pulses labelled as E, D or A are the E bits, D bits, and A bits respectively in the data frame. The synchronisation pulses are labelled with a V and are used to generate the required coding rule violations. In the timing diagram it is clear that for a given E, D or A symbol the pulse may appear on the SBUSRX_C or SBUSRX_D if a binary zero is being transmitted. For this reason both positive and negative pulses are drawn on the timing diagram to represent all the possible cases.

It is the coding rules incorporated within the state machine that determines whether a binary zero is transmitted as a positive or negative pulse. In the timing diagram the first coding rule violation is set with the L and F bits. The L bit following the F bit along with the Fa bit produce the second violation. A D bit or E bit could however produce a violation before the Fa bit if a binary zero is being coded. This procedure is repeated every frame. According to the coding rules a string of binary zeros should be coded such that the resulting pulses on the line alternate in polarity. In order to achieve this the state machine is required to remember the polarity of the last pulse transmitted. The state machine records the polarity of the last pulse transmitted using the variables LASTPULSE_POS and LASTPULSE_NEG. If a positive pulse is

transmitted the variable LASTPULSE_POS is set to logic 1 and LASTPULSE_NEG is cleared. If a negative pulse is transmitted the variable LASTPULSE_NEG is set to logic 1 and LASTPULSE_POS is cleared. This can be seen in the ASM chart.

The state number is identified at the top right hand of each state rectangle. State number 1 is identified as S1; state number 2 is identified as S2 and so on. The E and D bits are processed during state number 6. E and D bit data is read from text files (EBITS.txt and DBITS.txt) and coded by the state machine when the Test Sequence Generator sets the signals SEND_EBITS and SEND_DBITS active. During state number 6 if an INFO2 signal is turned on by the Test Signal Generator then the A bit is coded as a binary zero thus indicating INFO2. State 7 goes active in synchronism with the S-Bus count number 14. Here if a coding rule violation has not yet been achieved it is forced and the state sequence returns to state number 6 to repeat the process.

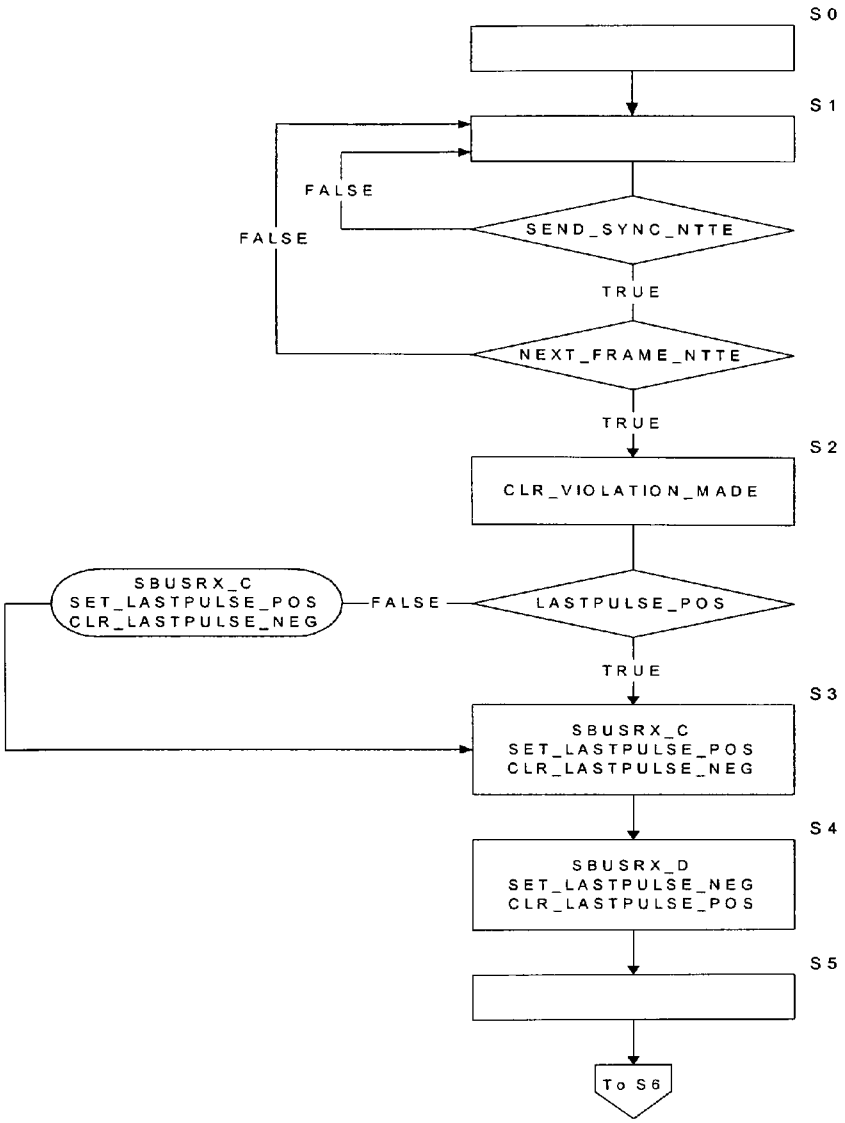


Figure 29a - The ASM chart for the NT to TE state machine.

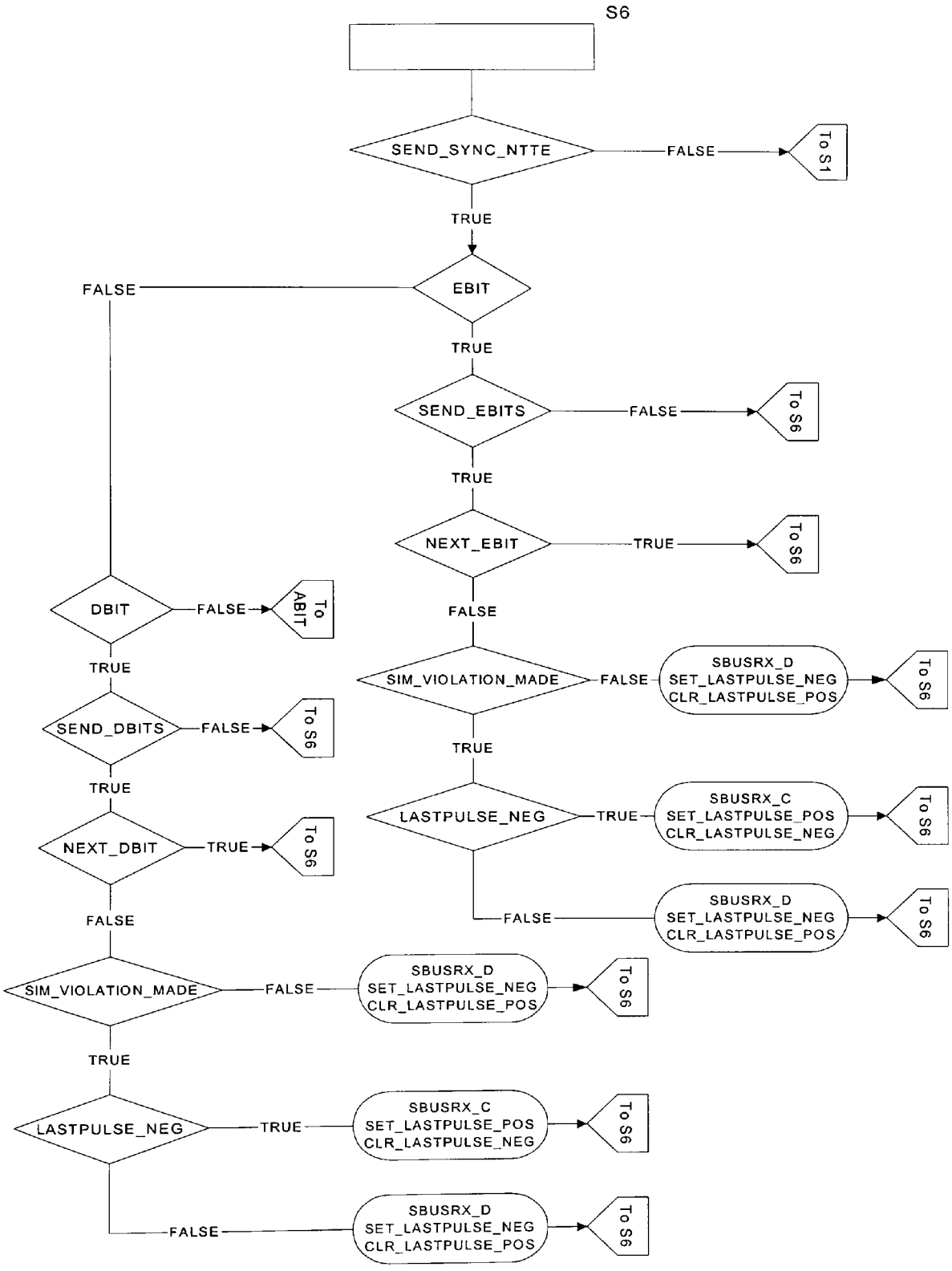


Figure 29b - The ASM chart for the NT to TE state machine.

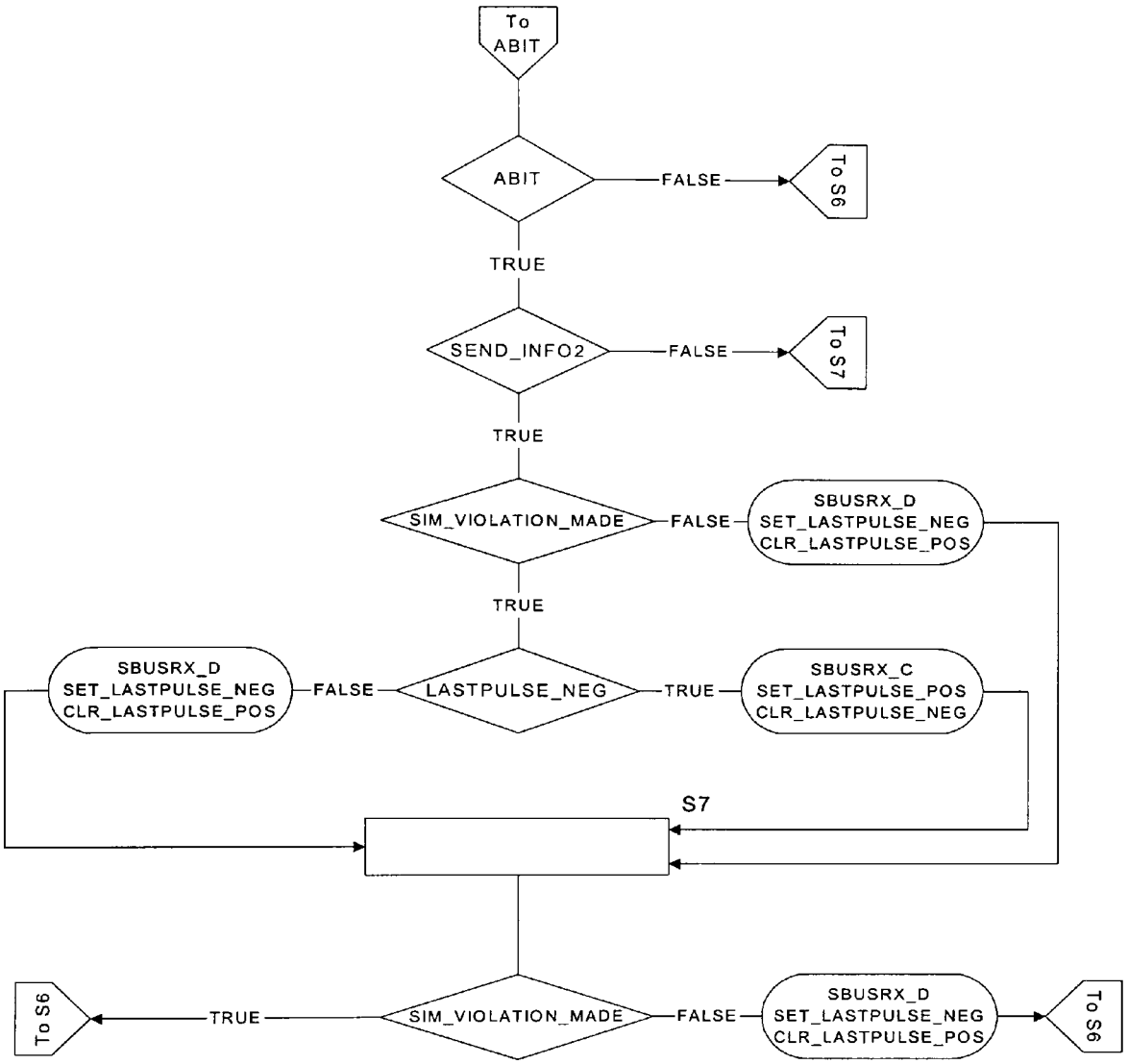


Figure 29c - The ASM chart for the NT to TE state machine.

4.4 The TE to NT Data Generator.

The TE to NT Data Generator produces the simulated S Bus signals in the TE to NT direction of data transmission. The Test sequence generator enables the generation of an INFO1 signal or synchronisation pulses. When the signal SYNC_INFO1 is set high an INFO1 is generated and when the signal SEND_SYNC_TENT is set high synchronisation frames are generated. The timing diagram describing the INFO1 generation procedure is presented in figure 30.

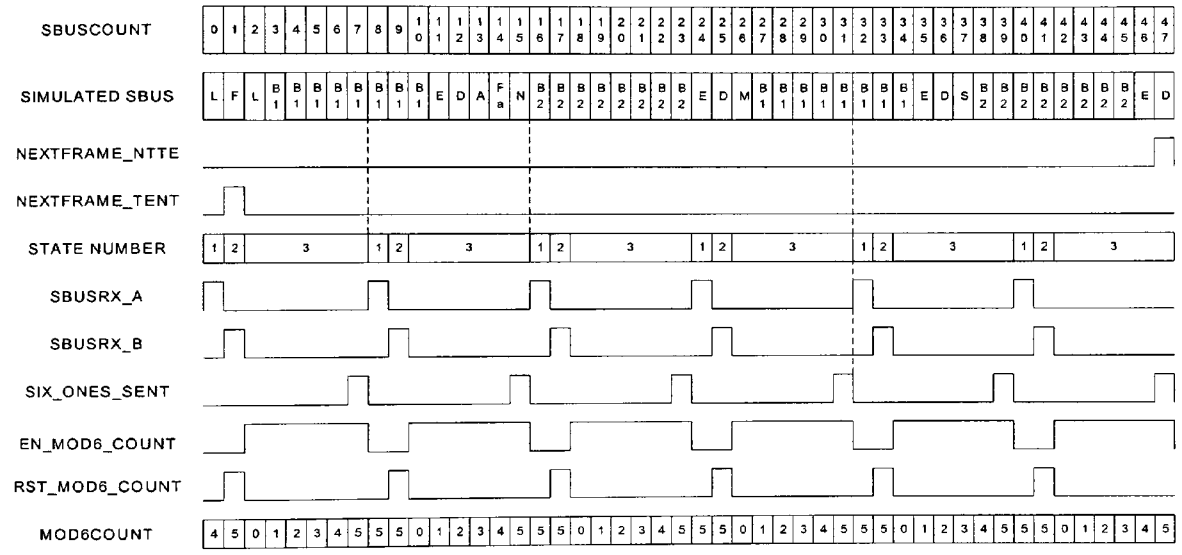


Figure 30 - Timing diagram for INFO1 generation.

The state machine algorithm employed by the TE to NT Generator is presented in figure 32. States number 1, 2, and 3 are entered when an INFO1 signal is generated. The output signals SBUSRX_A and SBUSRX_B simulate the signals from the analogue interface circuit. SBUSRX_A simulates a positive pulse and SBUSRX_B simulates a negative pulse. The state machine sets SBUSRX_A high during state number 1 represented by S1 in the ASM chart. This simulates a positive

pulse on the S Bus. State number 2 generates a pulse on SBUSRX_B to simulate a negative pulse on the S Bus. The state machine relies on a counter to count six pulses before repeating the process. This count sequence is shown on the timing diagram as MOD6COUNT. This count sequence is enabled during state number 3 with the EN_MOD6_COUNT. The count sequence is decoded to produce the signal SIX_ONES_SENT and the state machine returns to state number 1. The state machine continually loops through the states as shown in both the timing diagram and the ASM chart while the signal SEND_INFO1 remains high. When the Test Sequence Generator initiates generation of synchronisation frames, the signal SEND_INFO1 is returned low and SEND_SYNC_TENT is set high. The timing diagram for the synchronisation signal generation process is shown in figure 31.

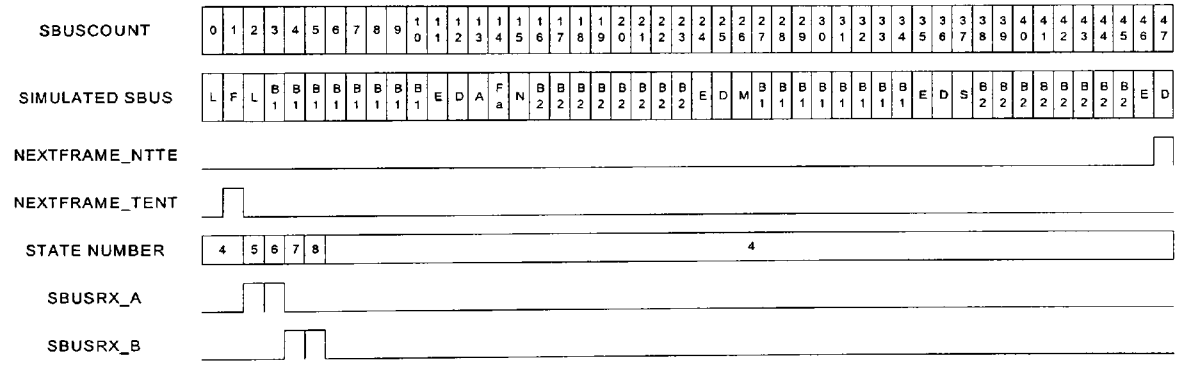


Figure 31 - The timing diagram for the Synchronisation signal generation process.

State numbers 4,5,6,7, and 8 are associated with the synchronisation frame generation. A synchronisation frame is generated every 250μs. The frame consists of a positive pulse in both L (SBUSCOUNT number 0) and F (SBUSCOUNT number 1) and a negative pulse in both L (SBUSCOUNT number 2) and B1 (SBUSCOUNT number 3) symbols. The state machine simply sets the signals SBUSRX _A and

SBUSRX_B high sequentially according to algorithm in figure 32. The one-hot state machine encoding process was employed during implementation. The process waits in state number 4 (S4) until the signal NEXTFRAME_NTTE goes active and then repeats the procedure.

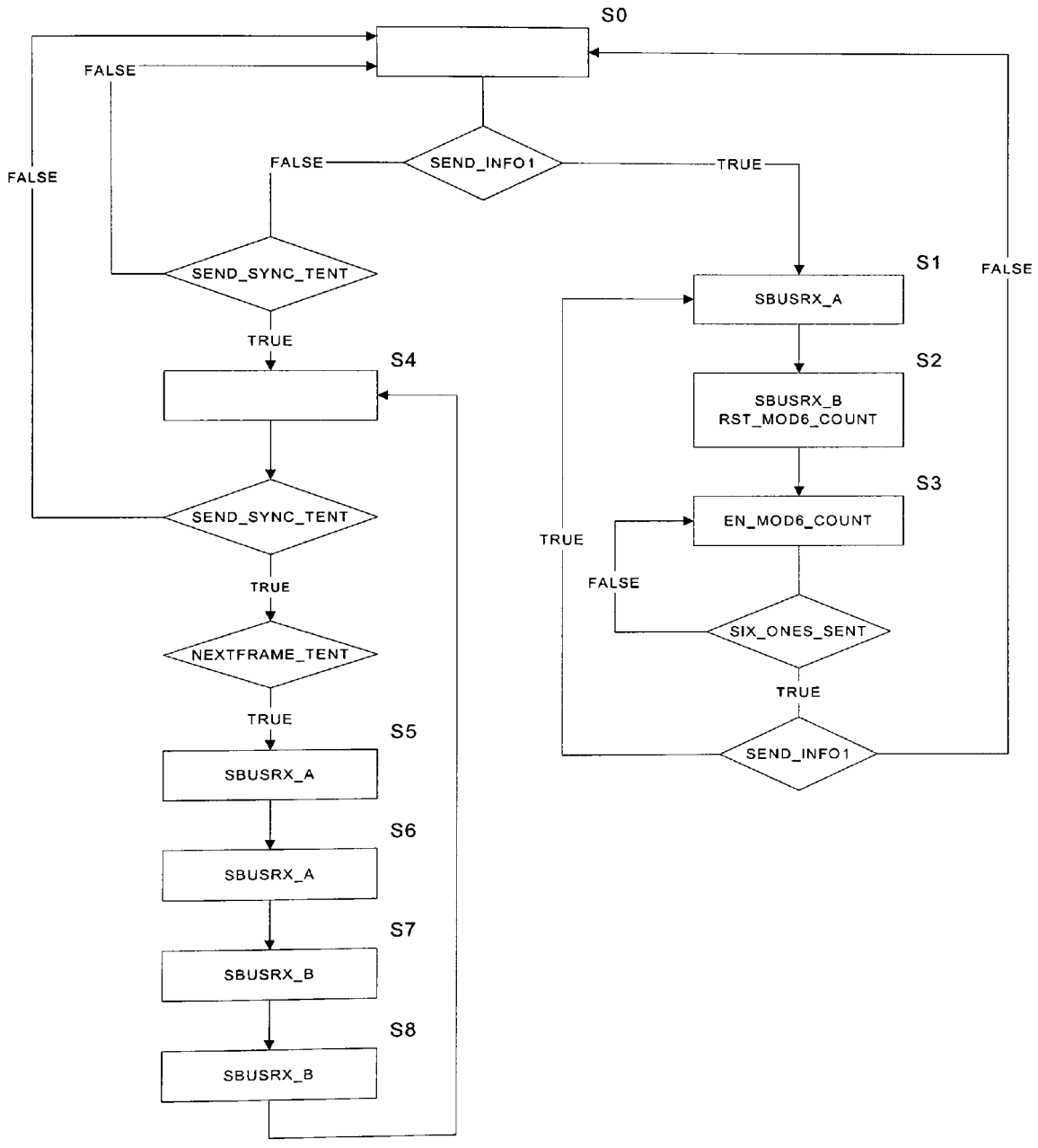


Figure 32 - The ASM chart for the TE to NT generator.

4.5 IOM-2 Monitor.

The IOM-2 Monitor performs a data logging function. The IOM-2 bus generated by the DUT is read at defined intervals and a text file is updated with D channel, E channel and Command/Indicate channel Information. The IOM-2 Monitor reads the IOM-2 bus count sequence generated by DUT. The count sequence is decoded and D and E channel information is extracted from the IOM-2 bus and written to a text file (EBITS_RESULT.txt and DBITS_RESULT.txt). The text file can be compared manually with the data encoded into the simulated S Bus from the EBITS.txt and DBITS.txt files. The Command/Indicate channel information is also extracted and written to a separate text file (IOM2_RESULT.txt). The timing diagram for the process is shown in figure 33.

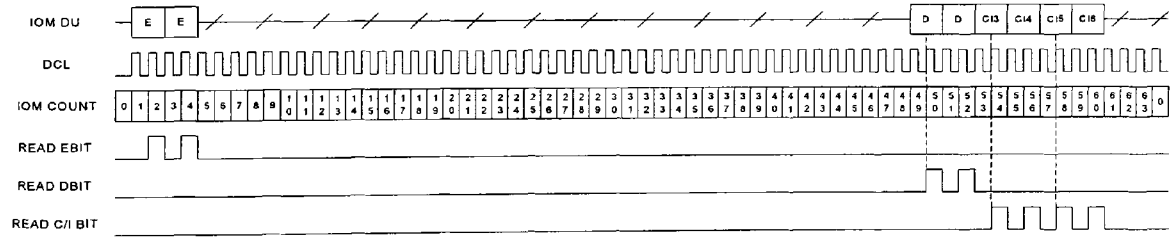


Figure 33 - The timing diagram for the IOM-2 Monitor.

The Command/Indicate channel bits are read when the signal READ C/I BIT is active. The file is opened when the Test bench starts running and the Command/Indicate channel data is logged to file over the whole test bench operating

time (8ms). The captured Command/Indicate data should reflect the INFO signal status on the simulated S Bus.

The D and E channel information will not be valid until both INFO4 and INFO3 signals are active on the ISDN S Bus. There is a data latency of 250μs from the time E and D channel information is inserted into the S Bus and that same data becoming valid on the IOM-2 bus. The signal LATCH_DBITS is produced by the DUT and indicates when D and E channel data is expected on the IOM-2 Bus. When the signals SEND_EBITS and SEND_DBITS are active, the IOM-2 monitor waits for the signal LATCH_DBITS and begins the D and E channel data-logging process.

Chapter 5

5 The Analysis of the Test Results.

The Mentor Graphics simulator, ModelSim was utilised to perform the digital simulation of the Firmware design. The Firmware design presented in chapter 3 will be referred to within this section as the device under test (DUT). A Test bench was written using VHDL. Test vectors were applied to the DUT and the Test bench was configured to monitor the resulting signals from the DUT. The simulators facility to produce a graphical representation of the resulting signals from the DUT was used during the analysis to verify the operation of each component within the design. In order to validate this design, it was necessary to compare signals with frequencies of MHz to signals with frequencies in kHz. The simulators zoom controls made selective viewing of waveform sections a relatively easy task. However, paper plots of waveform results were difficult to read. If a paper plot presenting an overall view of the test time from 0mS to 8ms is presented, the detail is compressed such that resolution is lost. If many pages are presented showing the detail over incremental time periods then the overall picture is difficult to read. For this reason, presentation of the test results is performed with text files that were written by the test bench during the simulation.

The analysis process focused on detailed verification of the S-Bus Interface circuit, IOM-2 Interface circuit and the IOM-2 Bus Clock Generator. The S-Bus synchronisation and PLL circuit validation focused on its ability to correctly achieve synchronisation with the simulated S Bus. The detailed operation of the S Bus synchronisation and PLL circuit has already been comprehensively validated in Report 2 and does not need further elaboration here.

The test bench resets the DUT at the beginning of the test sequence. Due to the synchronous nature of the reset circuitry associated with each flip-flop, a reset signal should be applied to each flip-flop for one whole clock period before a reset is achieved. In this case some of the flip-flops within the DUT are clocked with the 15.36MHz clock (C15_36MHZ) and the remaining flip-flops are clocked with either the 192kHz clock (MCLK_RX and MCLK_TX) or the 1.536MHz (DCL) clock. The circuitry that produces the 192kHz and 1.536MHz clock signals is clocked with the 15.36MHz clock. The reset signal holds the 192kHz and 1.536MHz clock signals at logic zero. Without the 192kHz and 1.536MHz clock signals, the circuitry driven by these clocks cannot be reset to a defined state. This can be overcome by modification of the DUT to allow asynchronous resetting. However this would not reflect the intended operation of the DUT and is thus undesirable.

The VHDL language offers the facility to initialise signals with a value within the VHDL code. This facility allows a designer to assign a logic 0 or a logic 1 to a signal when the circuit is reset. However, FPGA synthesis tools do not support this. If a signal is unassigned the simulator will assign an X to that signal until a valid result is produced. During the simulation the application of input signals will eventually produce valid output results and these results will propagate through the simulation until all signals in the design are assigned with a valid logic value. For this simulation the valid logic values are logic 1 (5V), logic 0 (0V) and Z (high impedance). When the reset signal is asserted, the signal activity following the reset will not be recorded in the result files while signals remain unassigned by the simulator. The test bench was designed this way to avoid confusion.

5.1 Analysis of the IOM-2 Bus Clock Generator.

The IOM-2 Bus Clock Generator circuit was verified by comparing of the waveforms produced by the simulator against the timing diagrams already presented in figure 17 and figure 18. In both cases it was deduced that the generated clock frequencies and duty cycles were correct.

5.2 Analysis of the S-Bus Interface functionality.

It is expected that the signals identified by the DUT should correspond to the simulated signals generated by the test bench. The firmware design was partitioned into components and presented in chapter 3. The S Bus Interface circuit is responsible for identifying the active INFO signals on the line. The test bench monitors the output from the circuit and writes a history of the recorded INFO signals to the INFO signal result file (INFO_SIG_RESULT.txt) presented in appendix C. The file is filled with lines of text containing the recorded signal information. A line of recorded data is presented below and an explanation on how to interpret it is given.

TIME = 8000005 ns INFO0 = 0 INFO2 = 0 INFO4 = 1 INFO0 = 0 INFO1 = 0 INFO3 = 1

Each line of data is divided into columns. Each column is separated with a space and a descriptor identifies the columns signal. For example, the first column is the time stamp introduced with the descriptor TIME. The time value is represented in nano-seconds and appears after the equal sign. The line of text above shows that the test bench logged the state of the S Bus signals at 8ms. The state of the signals in the

NT to TE direction of transmission (INFO0, INFO2, and INFO4) are recorded first followed by the state of the signals in the TE to NT direction (INFO0, INFO1 and INFO3). Each signal is named and has an associated signal status as indicated by either a 0 or a 1. A 0 indicates an inactive signal and a 1 indicates an active signal. A line of data is recorded every time there is a change in signal status. Firstly, the ability of the DUT to correctly identify the INFO signals on the simulated S Bus was validated. The timing diagram for the Test sequence generator has already been presented in figure 24 and will be referred to again here in order to facilitate this. The Test bench generates an INFO0 in both directions of transmission for 1ms. The INFO0 monitor circuits should indicate the presence of an INFO0 signal after 760.4μs. The INFO signal result file does not indicate this because at this point in the simulation the signals from the INFO0 monitor circuits are undefined. Careful investigation of the waveforms generated by the simulator confirmed that the DUT identified the presence of an INFO0 signal in both directions of data transmission at the correct time.

When the test bench begins generating the INFO1 signal at 1mS it is expected that the D.U.T. would identify this after five successive INFO1 signals. The timing diagram in figure 10 states that a time period of 215.8μs is required to identify an INFO1 signal. The INFO1 monitor will hunt continually for the presence of an INFO1 signal. When identification of the INFO1 signal is made the INFO1 monitors internal counter circuitry is reset. The timing diagram does not account for the extra time that this action adds to the signal identification process. The worst-case time for the INFO1 signal identification would include an extra 52μs. The INFO1 signal transmission is enabled by the Test bench at 1mS therefore it is expected that the D.U.T would indicate the presence of an INFO1 signal at (1ms + 215.8μs + 52μs =

1.268ms). The signal is recorded as active by the test bench at 1.268ms. The graphical result produced by the simulator was checked and it was confirmed that the D.U.T correctly identified the presence of an INFO1 after five consecutive INFO1 signals. The test bench disables the transmission of the INFO1 signal after 2ms. The INFO1 monitor should read binary ones from the simulated S-Bus until the test bench enables the transmission of synchronisation pulses after 4ms. The INFO1 monitor would be expected to cancel the INFO1 signal indication after five INFO1 signals. The last INFO1 signal would still have to propagate through the system making the total time (6 x 41.6μs) + 10.4μs = 254.8μs. The result file should indicate this change at 2ms + 254.8μs = 2.2548ms. The actual time logged is 2.26ms. The extra 5.2μs was found to be associated with the time to propagate the last INFO1 signal through the system. The waveforms produced by the simulator relating to the INFO1 monitor operation were checked manually and correct operation was verified.

The test bench begins generating an INFO2 signal at 2ms. It is expected that the D.U.T should firstly establish synchronisation before enabling the INFO2 detection circuit. The time to establish synchronisation is dependent upon the successful identification of three consecutive sets of coding rule violations. The coding rules have already been presented in report number two. Based on the current simulation, synchronisation is established with the simulated S Bus within 833μs. Detailed analysis of the simulation waveforms verified that the synchronisation circuit operated as expected. An active INFO4 signal indicates synchronisation with the NT to TE direction of data transmission. The result file indicated that INFO4 went active at 2.834ms (2ms + 834μs) as expected. There is a delay in recording a change in signal status due to the time required to identify the change in signal status and the time required to initiate a result file write operation.

Once synchronisation with the NT to TE direction is achieved the INFO2 monitor circuit is enabled. When the INFO2 monitor circuit begins hunting it is expected that identification of the INFO2 signal should occur within four S Bus frames. The timing diagram in figure 13 indicates that INFO2 should be identified within 585.8μs. Therefore the total detection time for an INFO2 signal is approximately 2ms + 834μs + 585.8μs = 3.4ms. The result file recorded an active INFO2 signal at 3.5ms. The graphical waveforms produced by the simulator were analysed in detail and it was verified that the INFO2 detection circuit correctly identified INFO2 within four frames of being enabled. The test bench enables the generation of synchronisation frames in both directions of data transmission at 4ms. It is expected that the DUT should cancel the INFO2 signal indication within 580.6μs as explained in 3.4. The result file indicates that the INFO2 indication was cancelled at 4.581ms. The signal waveforms associated with the INFO2 detection circuit were investigated manually and correct operation of the circuit was verified.

Synchronisation with the data frames flowing in the TE to NT direction is also expected after three consecutive coding rule violations have been detected. The waveforms produced by the simulator were checked to verify that this was the case. The INFO0 monitor circuit connected to the TE to NT direction of data flow was also operating at this point in time. While the INFO0 signal is active the DUT does not indicate INFO3. As soon as the INFO0 signal was cancelled the INFO3 signal indication was produced as expected. This event occurred at 4.778ms and is recorded in the INFO signals result file. The results of the verification exercise are summarised in figure 34. Here the waveforms produced by the simulator have been reproduced. The waveforms are not to scale however the illustration summarises the detection times of each INFO signal in graphical form.

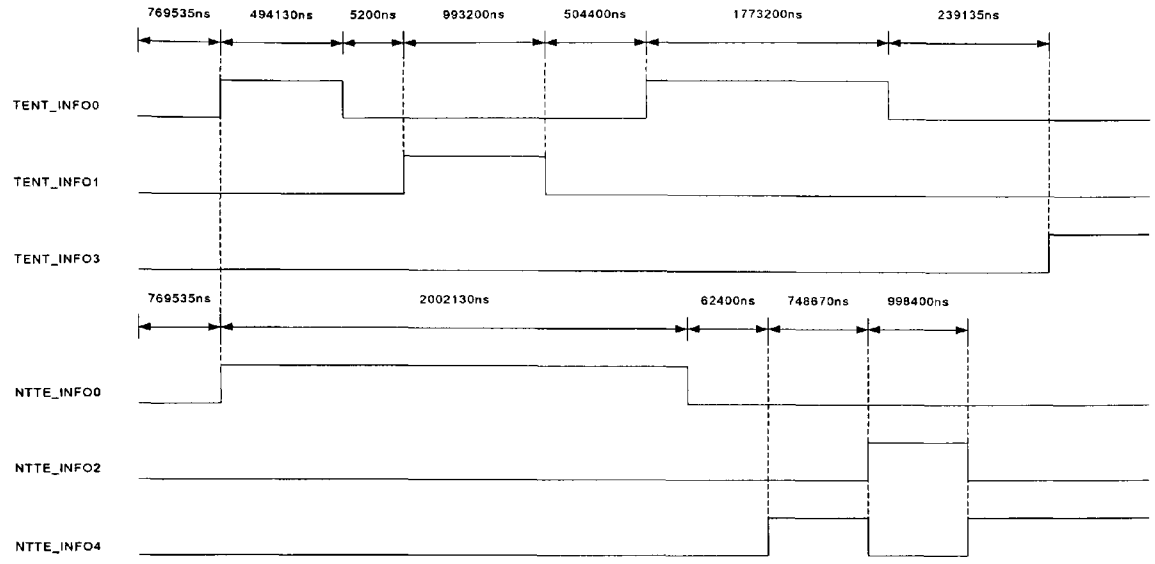


Figure 34 - The simulation results for the S Bus Interface Circuit.

5.3 The analysis of the IOM-2 Bus Interface functionality.

The D.U.T transmits both D and E channel information and INFO signal status to the microcontroller via the IOM-2 bus. It is the responsibility of the IOM-2 Interface circuit to maintain an indication of the currently active INFO signal on the S bus. The INFO signal status is encoded within the Command/Interface channel of the IOM-2 bus according to table 1. The test bench monitors the resulting IOM-2 bus signals and writes the contents of the Command/Interface channel to the IOM-2 results text file (IOM2_RESULT.txt). The text file is structured according to the example shown below.

E0: 0 E1: 1 D0: 0 D1: 0 C/I0_3: 1 C/I0_4: 1 C/I0_5: 1 C/I0_6: 1 TIME: 7924215 ns FRAME NUMBER: 63

Each line of text represents the IOM-2 data from one frame. The text line records a time stamp and an IOM-2 frame number. The test bench records the E and D channel information along with the command/Indicate channel status and writes this information to the text file (IOM2_RESULT.txt). The result file can be seen in appendix D. A 1 written to the result file represents logic one and a 0 written to file represents logic zero. The above text line represents the IOM-2 data at 7.924ms and is frame number 63. The E and D channel information is shown and the Command/Indicate channel information indicates that an INFO3 and an INFO4 is active. Once the simulation is complete the IOM-2 result file will have recorded 8mS of data. The result file was manually checked to ensure that the correct information is sent to the Microcontroller. The following section presents the analysis of the IOM-2 data transmitted by the DUT within the command/Indicate channel for both directions of data transmission.

Analysis of the Command/Indicate channel on the IOM-2 bus.

The active signal on the S Bus is first identified by the S Bus Interface circuit then passed to the IOM-2 Interface circuit for encoding within the Command/Indicate channel. This analysis exercise had two objectives. Firstly, it was verified that the IOM-2 Command/Indicate channel was updated at the correct time with the active INFO signal as identified by the S Bus interface. Secondly, it was verified that the Command/Indicate channel was encoded correctly according to table 1. The active NT to TE signal indication is encoded within the Command/Indicate channel bits C/I0_3 and C/I0_4 of the IOM-2 bus. The IOM-2 Interface circuit should encode an undefined signal in the Command/Indicate channel until a valid signal can be reliably

identified by the S Bus Interface Circuit. The IOM-2 results file was checked to verify that this was the case.

The test bench updates the IOM-2 results file with the Command/Indicate channel codes at the end of each IOM-2 frame. The simulation waveforms were investigated in detail and it was verified that signal changes were reflected in the Command/Indicate channel of the next IOM-2 frame. It was also verified that the IOM-2 Interface Circuit maintained the Command/Indicate channel code associated with an active INFO signal while the INFO signal was active. The following section reviews the information written to the IOM-2 results file by the Test bench. The analysis firstly reviews the signals flowing from the TE toward the NT. The analysis then focuses on the NT to TE direction of signal flow.

Analysis of the signals flowing in the TE to NT direction.

The signals flowing in the TE to NT direction are encoded within the C/I0_5 and C/I0_6 bits of the Command/Indicate channel. As already presented in figure 34 the S Bus Interface circuit recognised the presence of an INFO0 after 769535ns in both directions of data flow. The IOM-2 result file recorded an active INFO0 signal at 805415ns for both directions of data transmission. The encoding was correct and manual checking of the waveforms produced by the simulator verified that the Command/Indicate channel was updated in the next IOM-2 frame following the indication of INFO0 by the S Bus Interface circuit. Prior to identification of the INFO0 signal, the IOM-2 result file indicated an undefined signal for both directions of data transmission as expected.

According to figure 34, the S Bus Interface circuit identifies the INFO1 signal at 1268865ns (769535ns + 494130ns + 5200ns) and cancels the INFO1 signal indication at 2262065ns (769535ns + 494130ns + 5200ns + 993200ns). The Command/Indicate channel bits C/I0_5 and C/I0_6 bits of the IOM-2 result file are updated to reflect the existence of an INFO1 signal at 1304615ns. The IOM-2 results file indicates a cancellation of the INFO1 signal at 2303145ns (Frame number 18) when the Command/Indicate channel is updated to indicate an undefined signal. According to figure 34, the S Bus Interface circuit identified the existence of an INFO0 signal again at 2766465ns (769535ns + 494130ns + 5200ns + 993200ns + 504400ns). This signal change is updated in the IOM-2 results file at 2807545ns (Frame number 22). Figure 34 indicated that the S Bus Interface circuit cancelled the INFO0 signal indication at 4539665ns (769535ns + 494130ns + 5200ns + 993200ns + 504400ns + 1773200ns). The IOM-2 result file correctly updated the C/I0_5 and C/I0_6 bits to indicate an undefined signal at 4554615ns (Frame number 36). The S Bus Interface circuit finally identified the INFO3 signal generated by the Test bench at 4778800ns (4539665ns + 239135ns). The IOM-2 results file updated the Command/Indicate channel at 4804215ns (Frame number 38). The waveforms produced by the simulator were checked and it was verified that the updating of the IOM-2 bus was performed at the correct times in all cases.

Analysis of the signals flowing in the NT to TE direction.

The signals flowing in the NT to TE direction are encoded within the C/I0_3 and C/I0_4 bits of the Command/Indicate channel. The IOM-2 result file correctly indicated the existence of an undefined signal before 805415ns (Frame number 6) in

the same way as the C/I0_5 and C/I0_6 bits. Figure 34 indicates that the INFO0 signal is cancelled at 2771665ns (769535ns + 2002130ns). The IOM-2 results file indicates that the C/I0_3 and C/I0_4 bits were encoded as an undefined signal correctly at 2807545ns (Frame number 22). According to Figure 34 the S Bus Interface synchronises with the NT to TE direction of data flow at 2834065ns (769535ns + 2002130ns + 62400ns). This is interpreted as an INFO4 signal by the IOM-2 Interface circuit and the C/I0_3 and C/I0_4 bits are encoded to reflect this. The IOM-2 result file indicates that both C/I0_3 and C/I0_4 bits are set to logic one at 2932345ns (Frame number 23). The S Bus Interface cancels the INFO4 signal indication and sets the INFO2 signal indication at 3582735ns (769535ns + 2002130ns + 62400ns + 748670ns). The IOM-2 results file records this signal change at 3681015ns (Frame number 29). The Command/Indicate channel was encoded as C/I0_3 = 1 and C/I0_4 = 0 to indicate the INFO2 signal correctly. The S Bus Interface cancels the INFO2 signal Indication and sets the INFO4 signal Indication at 4581135nS (769535nS + 2002130ns + 62400ns + 748670ns + 998400ns). The IOM-2 Result file was updated at 4679415ns (Frame no 37) to encode the Command/Indicate channel as C/I0_3 = 1 and C/I0_4 = 1. This indication of INFO4 was maintained for the remainder of the simulation as expected. The simulation waveforms were checked in all cases and it was verified that changes in signal status caused an update of the Command/Indicate channel in the next IOM-2 bus.

Analysis of the D and E channels.

During the simulation, D and E channel information was read from text files (DBITS.txt and EBITS.txt) by the test bench and transmitted on the simulated S Bus in the NT to TE direction. The analysis procedure verified that the D and E channel information was correctly extracted by the S Bus Interface circuit and then transmitted on the IOM-2 bus by the IOM-2 Interface circuit. The test bench monitored the IOM-2 bus and read the logic state of the D and E channel bits. The resulting D and E channel information was written to two separate result files (DBITS_RESULT.txt and EBIT_RESULT.txt) by the test bench. The analysis procedure then involved a manual comparison of the two sets of files. The files are presented in appendix E. The result of the comparison verified that the information was correctly retransmitted on the IOM-2 bus.

Chapter 6

6 Conclusion and Further Work.

6.1 Conclusions.

This chapter reviews the achievements of the study in relation to the objectives and conclusions are drawn. The main objective was to demonstrate how Field Programmable Gate Array (FPGA) technology could be used to improve on the typical Integrated System Digital Network (ISDN) protocol analyser design approach adopted by Industry. The study focused on the development of FPGA Firmware circuitry to implement the interfacing between the ISDN S bus and the Microcontroller. The aim was to design a low cost solution with the flexibility to change the design at a later stage without incurring substantial cost. The aims included implementing a solution that undermined the dependence on a single ASSP manufacturer to supply components for production.

The design realisation succeeded in providing an ISDN passive monitor solution that was an improvement over those implemented with commercially available ASSPs. The implemented design successfully demonstrates the advantages of using FPGA technology in low volume test equipment applications. The following conclusions can be drawn from the study.

1. The digital circuitry required for the ISDN monitor was integrated within one Xilinx FPGA. The use of the FPGA makes the requirement for the two ISDN ASSPs redundant. The extra features offered by the ISDN ASSPs that are not required in ISDN monitoring applications add no value to the product for the cost incurred. Extra features include the S Bus transmission circuits. The

complexity of the transmission circuit has already been demonstrated in Report number two. In the typical implementation where two ISDN ASSPs are adopted in the design, the S Bus transmitter circuits are not used but may be consuming power. In the case of the improved design, the implementation concentrated on providing only those features required by the application. The FPGA Computer Aided Engineering (CAD) design tools allowed the identification of a suitably sized FPGA for the design. The implemented design consumed 73% of the resources offered by a Xilinx 4010 FPGA. This allows for future expansion while providing the lowest cost solution to the problem. The study clearly demonstrates the advantages of the FPGA design approach over the conventional design approach.

2. The design process partitioned the FPGA Firmware into functional blocks. The use of the Very High Speed Hardware Description Language (VHDL) to achieve the design implementation facilitated design portability. Functional blocks written in VHDL may be used in different projects. For example, the S Bus synchronisation and Phase Locked Loop (PLL) circuit designed and implemented for use in the Bit Error Rate tester (BERT) project (Report number two) was used again here in the ISDN Monitor project. The design effort given to this task during the BERT project was not required during this project. The functional design blocks such as the IOM-2 Interface circuit and the S Bus Interface circuit may also be transported to other projects. The careful identification and partitioning of key design components required for ISDN test applications allows for the development of a library of reusable components. Existing components may be modified or new components may be added as required. Commitment to the FPGA design approach has the

advantage of total ownership of design data. New designs can be realised in a much shorter time than by embarking on the more expensive ASSP design route. The FPGA Firmware can be updated in the field to provide the customer with upgrades as the design is modified. This achieves the objective to provide a flexible design solution that may be adapted to meet the requirements of a changing market.

3. Manufacturers such as Xilinx and Atmel both produce a family of compatible FPGAs that have identical footprints and programming specifications. The logic design was successfully targeted at a Xilinx FPGA but may also be targeted at an Atmel device so that manufacturer independence is achieved. The logic design may of course be targeted at FPGAs from other manufacturers.

6.2 Future work.

The project work successfully achieved the design and simulation of the circuits to meet the objectives outlined in chapter one. The following tasks have been identified to progress to work.

1 The 68302 Microcontroller manufactured by Motorola semiconductors has been identified as a suitable choice to perform the analysis of the Link Access Protocol for the D channel (LAP-D) signals for layer two and layer three. The 68302 microcontroller has four integrated hardware HDLC circuits. The 68302 microcontroller supports the IOM-2 interface standard and would interface directly to the current FPGA design. Software development to perform the layer 2 and layer 3 signal analysis would be required.

2 HDLC controllers have already been implemented as logic cores for FPGA applications. The features offered by the FPGA may be extended to include LAP-D analysis by incorporating HDLC controller cores within the design. The interfacing system between the FPGA and the Microcontroller may be implemented as either the industry standard Serial Peripheral Interface (SPI) bus or the IOM-2 bus. This would allow an increased Microcontroller choice.

References

[1] International Telecommunications Union (ITU-T), Recommendation I.430, Basic User-Network Interface – Layer 1 Specification, 1993.

[2] J.Dunlop and D.G. Smith, Telecommunications Engineering, Stanley Thornes (Publishers) Ltd, ISBN 0-7487-4044-9.

[3] International Telecommunications Union (ITU-T), Recommendation Q.921, ISDN user-network interface – Data link layer specification. 09/1997.

[4] International Telecommunications Union (ITU-T), Recommendation Q.931, ISDN user-network interface layer specification for basic call control. 05/1998.

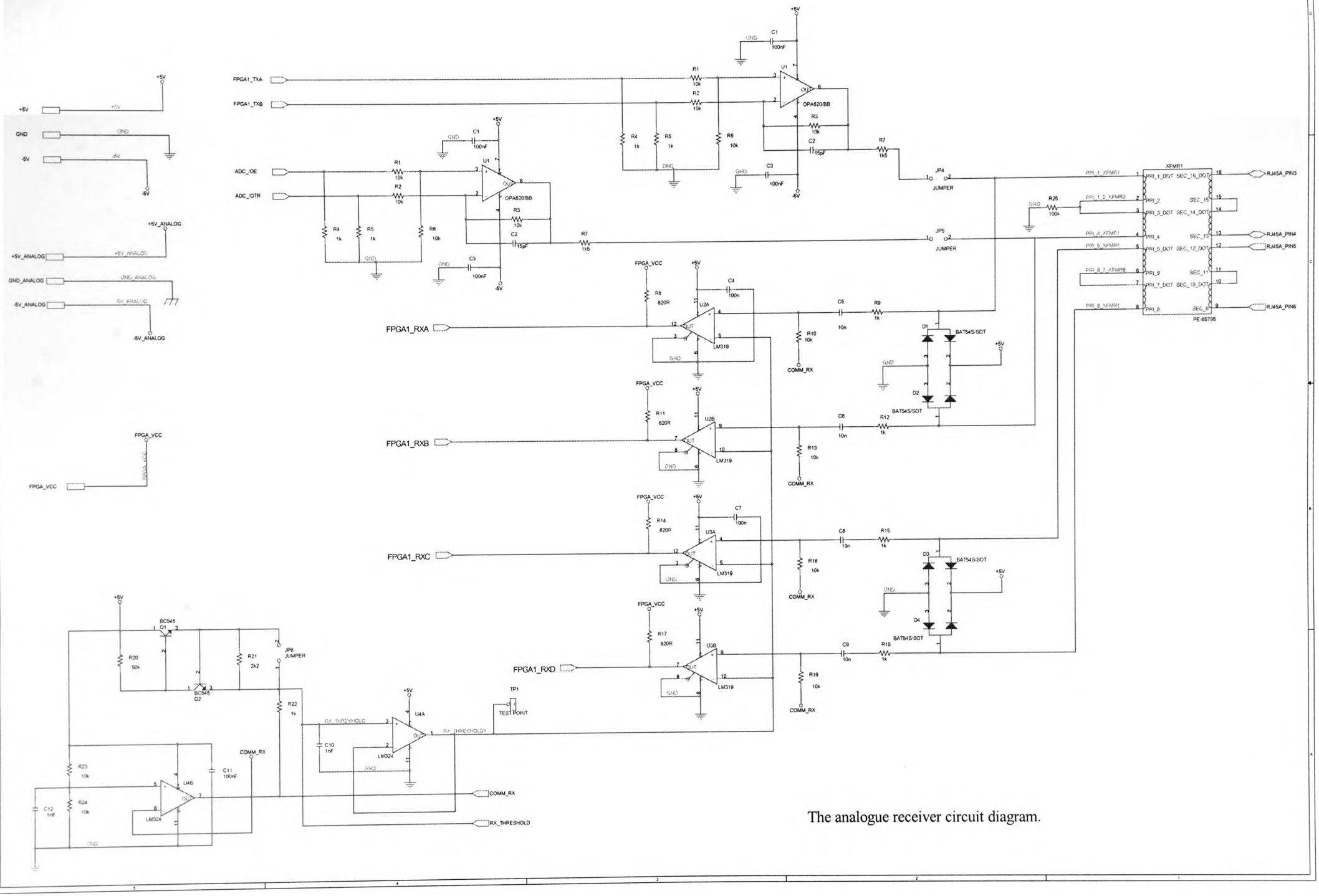
[5] Infineon Technologies, Data Sheet for the PEB3081 S bus interface. Rev 1.4. Feb 2003.

[6] Motorola Semiconductors, Data sheet for the MC145574DE S/T-interface, Rev 5, 28/6/1999.

[7] Siemens, The IOM-2 Interface Reference Guide, Version 01.90, March 1991.

Appendix A

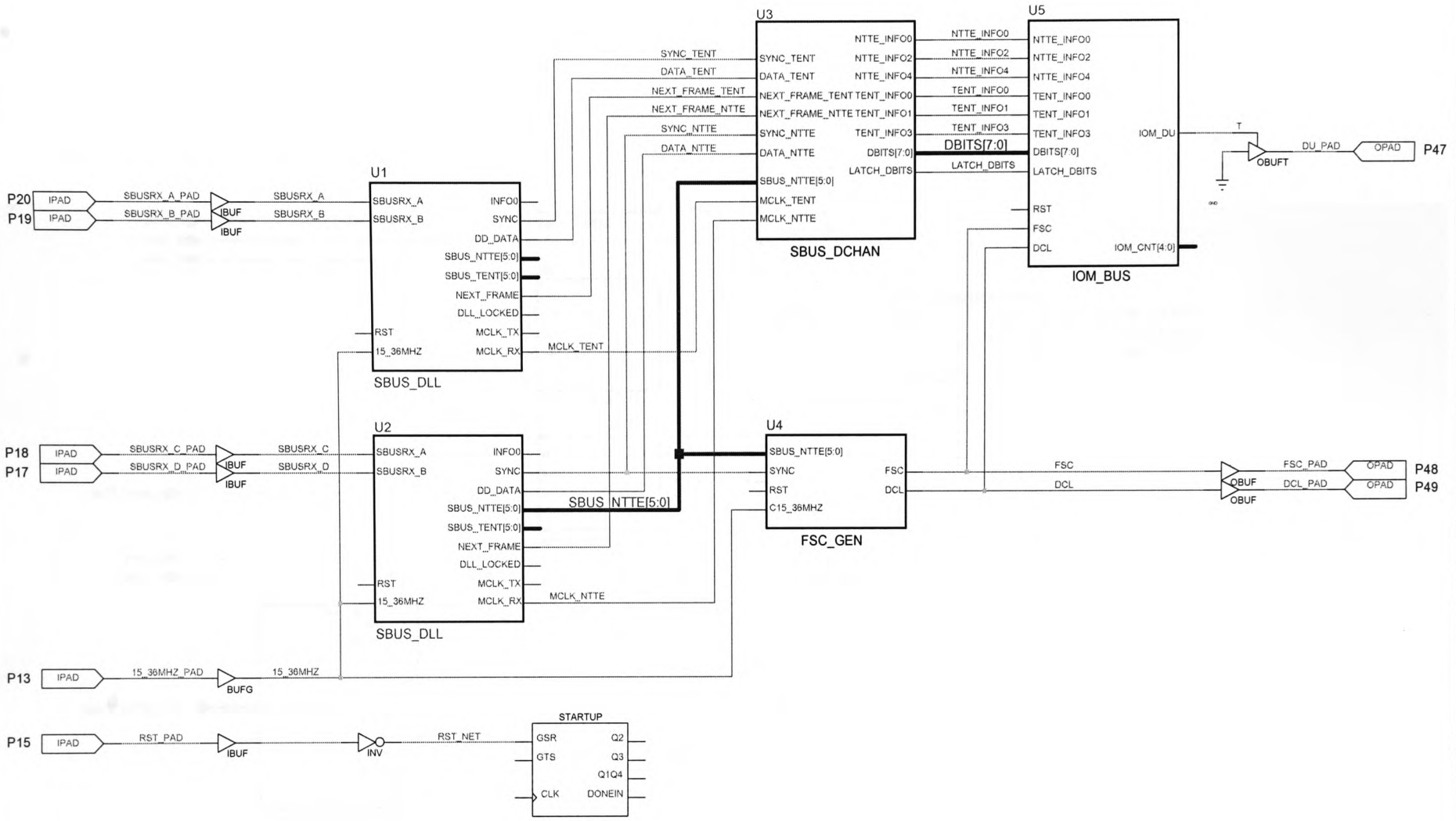
The analogue receiver circuit diagram.



The analogue receiver circuit diagram.

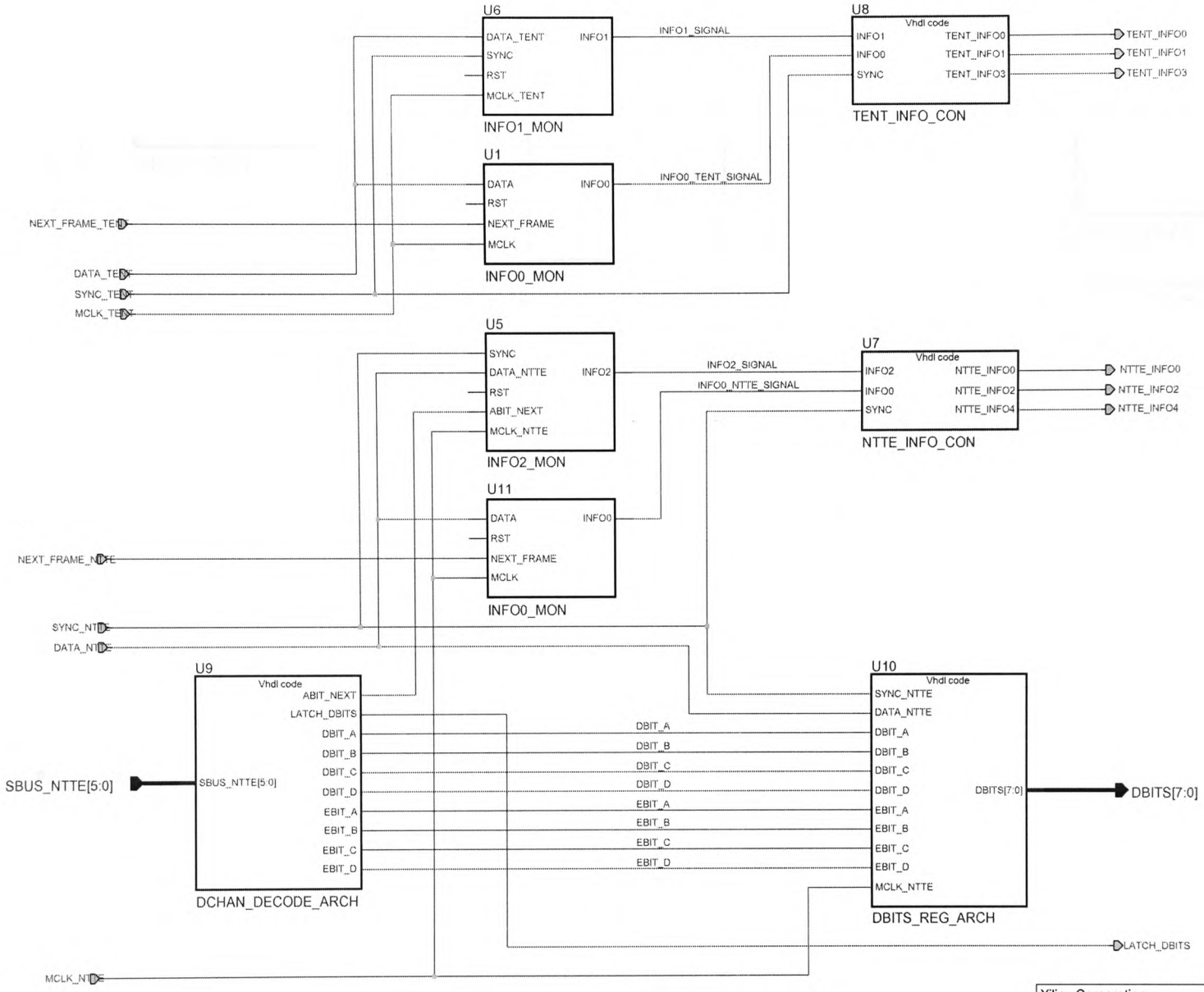
Appendix B

The FPGA firmware block diagram.



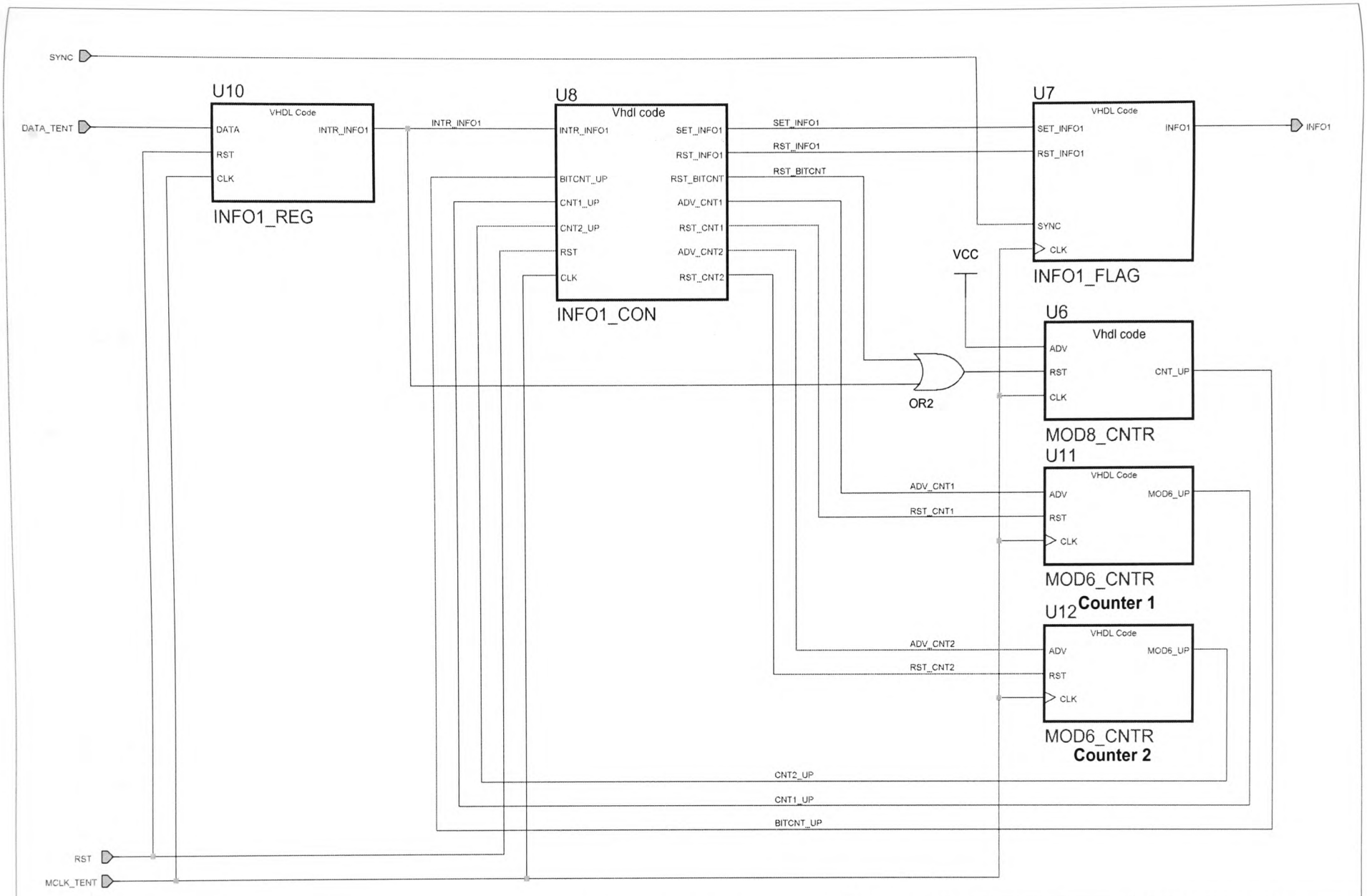
Appendix B, Figure 1.

Xilinx Corporation	Project: SBUS_MON
2100 Logic Drive	Macro: SBUS_MO1
San Jose, CA 95124	Date: 9/18/01
Date Last Modified: 9/20/1	



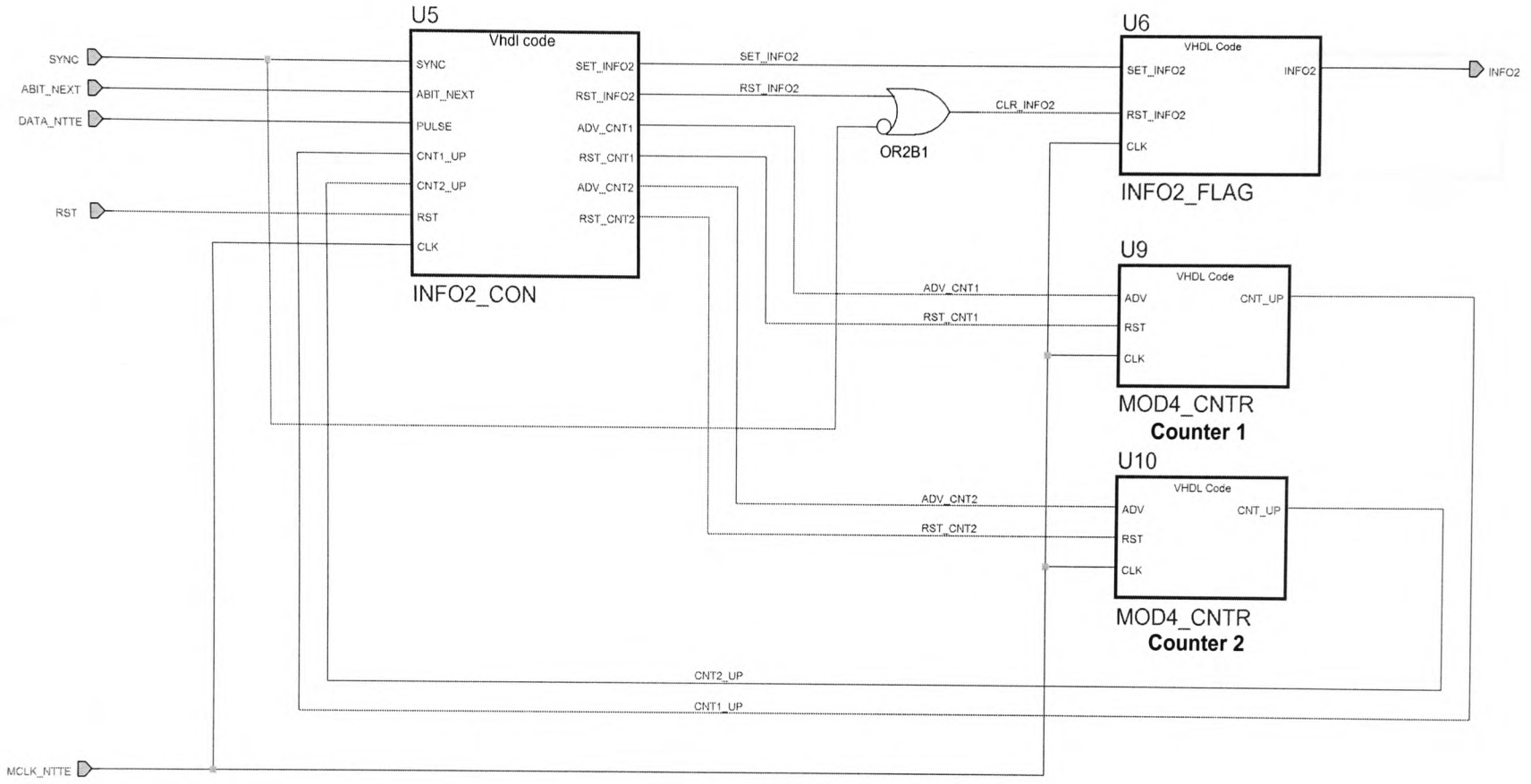
Appendix B, Figure 2

Xilinx Corporation	Project: SBUS_MON
2100 Logic Drive	Macro: SBUS_DCHAN
San Jose, CA 95124	Date: 9/18/01
Date Last Modified: 9/20/1	



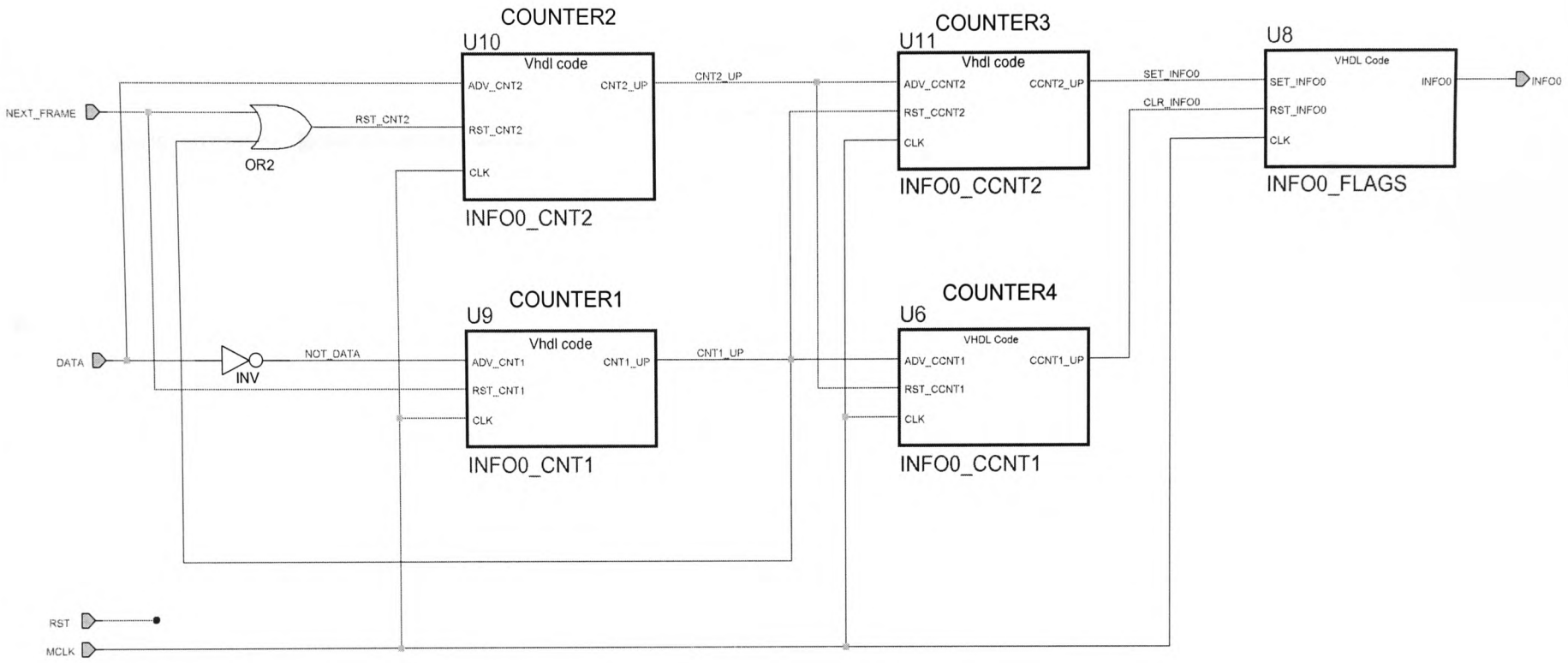
Appendix B, Figure 3

Xilinx Corporation	Project: SBUS_MON
2100 Logic Drive	Macro: INFO1_MON
San Jose, CA 95124	Date: 9/18/01
Date Last Modified: 9/20/1	



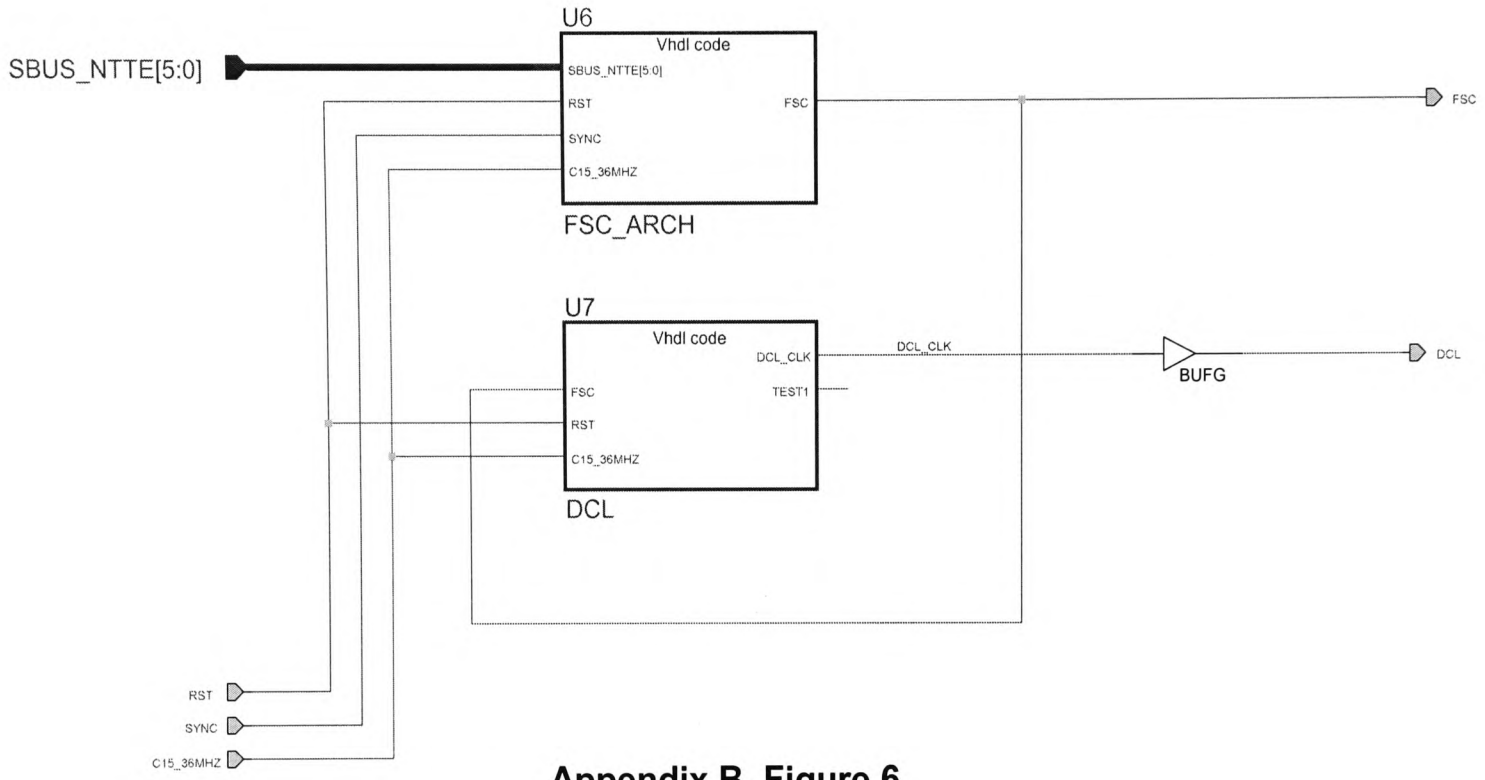
Appendix B, Figure 4

Xilinx Corporation	Project: SBUS_MON
2100 Logic Drive	Macro: INFO2_MON
San Jose, CA 95124	Date: 9/18/01
Date Last Modified: 9/20/1	



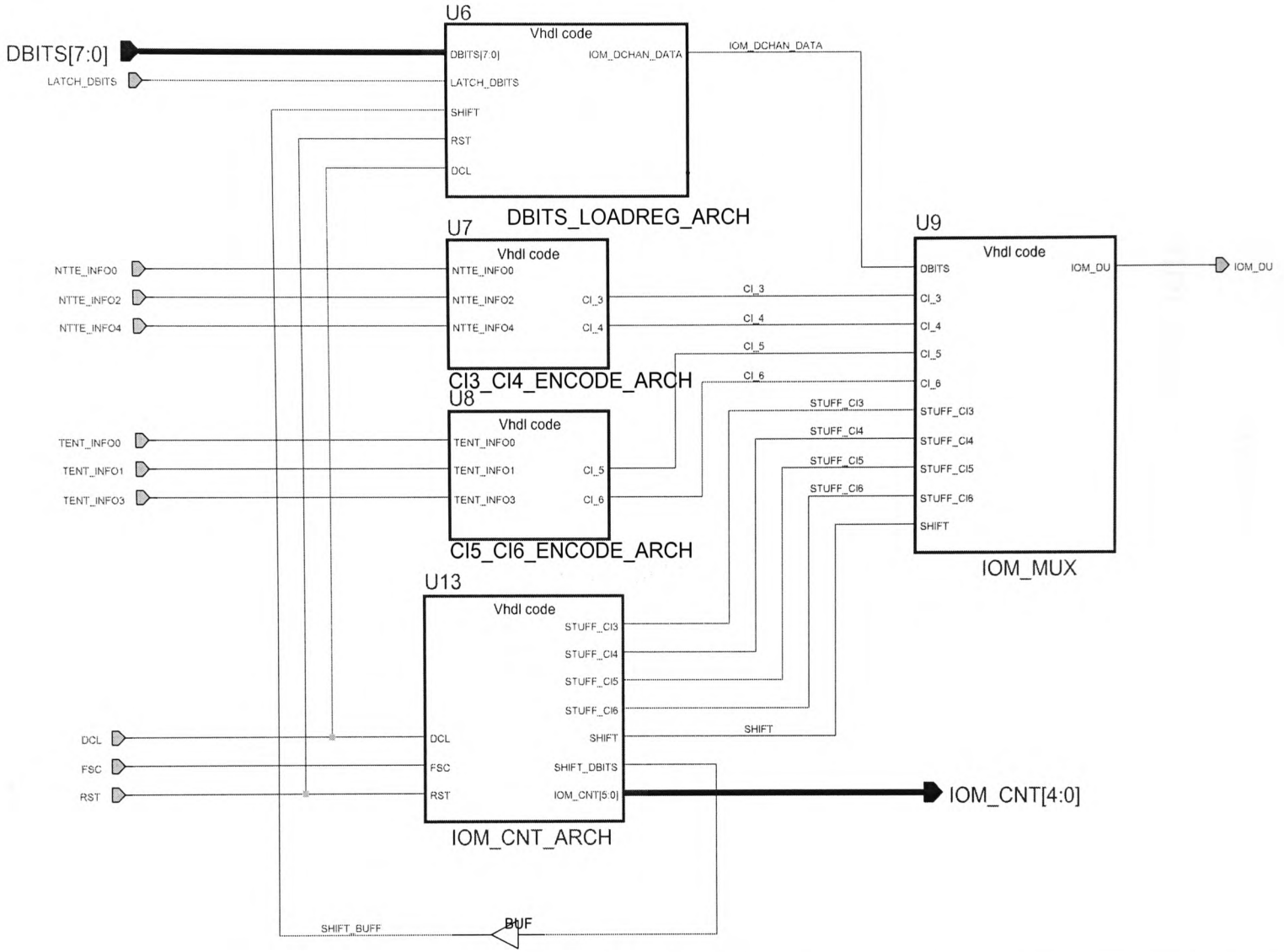
Appendix B, Figure 5

Xilinx Corporation	Project: [None]
2100 Logic Drive	Macro
San Jose, CA 95124	Date: 9/18/01
Date Last Modified:	



Appendix B, Figure 6

Xilinx Corporation	Project: [None]
2100 Logic Drive	Macro
San Jose, CA 95124	Date: 9/18/01
Date Last Modified:	



Appendix B, Figure 7

Xilinx Corporation	Project: SBUS_MON
2100 Logic Drive	Macro: IOM_BUS
San Jose, CA 95124	Date: 9/18/01
Date Last Modified: 9/20/1	

Appendix C

VHDL results file – info_sig_results.txt.

The INFO signals result file (INFO_SIG_RESULT.txt)

TIME = 1263730 ns INFO0 = 1 INFO2 = 0 INFO4 = 0 INFO0 = 0 INFO1 = 0 INFO3 = 0
TIME = 1268930 ns INFO0 = 1 INFO2 = 0 INFO4 = 0 INFO0 = 0 INFO1 = 1 INFO3 = 0
TIME = 2262130 ns INFO0 = 1 INFO2 = 0 INFO4 = 0 INFO0 = 0 INFO1 = 0 INFO3 = 0
TIME = 2766530 ns INFO0 = 1 INFO2 = 0 INFO4 = 0 INFO0 = 1 INFO1 = 0 INFO3 = 0
TIME = 2771730 ns INFO0 = 0 INFO2 = 0 INFO4 = 0 INFO0 = 1 INFO1 = 0 INFO3 = 0
TIME = 2834130 ns INFO0 = 0 INFO2 = 0 INFO4 = 1 INFO0 = 1 INFO1 = 0 INFO3 = 0
TIME = 3582800 ns INFO0 = 0 INFO2 = 1 INFO4 = 0 INFO0 = 1 INFO1 = 0 INFO3 = 0
TIME = 4539730 ns INFO0 = 0 INFO2 = 1 INFO4 = 0 INFO0 = 0 INFO1 = 0 INFO3 = 0
TIME = 4581200 ns INFO0 = 0 INFO2 = 0 INFO4 = 1 INFO0 = 0 INFO1 = 0 INFO3 = 0
TIME = 4778865 ns INFO0 = 0 INFO2 = 0 INFO4 = 1 INFO0 = 0 INFO1 = 0 INFO3 = 1
TIME = 8000005 ns INFO0 = 0 INFO2 = 0 INFO4 = 1 INFO0 = 0 INFO1 = 0 INFO3 = 1

Appendix D

VHDL results file – iom2_result.txt.

The IOM-2 Result file (IOM2_RESULT.txt).

E0: 0 E1: 0 D0: 0 D1: 0 C/I0_3: 0 C/I0_4: 0 C/I0_5: 0 C/I0_6: 0 TIME: 51350 ns FRAME NUMBER: 0
E0: 0 E1: 0 D0: 0 D1: 0 C/I0_3: 0 C/I0_4: 0 C/I0_5: 0 C/I0_6: 0 TIME: 182065 ns FRAME NUMBER: 1
E0: 0 E1: 0 D0: 0 D1: 0 C/I0_3: 0 C/I0_4: 0 C/I0_5: 0 C/I0_6: 0 TIME: 306215 ns FRAME NUMBER: 2
E0: 0 E1: 0 D0: 0 D1: 0 C/I0_3: 0 C/I0_4: 0 C/I0_5: 0 C/I0_6: 0 TIME: 431015 ns FRAME NUMBER: 3
E0: 0 E1: 0 D0: 0 D1: 0 C/I0_3: 0 C/I0_4: 0 C/I0_5: 0 C/I0_6: 0 TIME: 555815 ns FRAME NUMBER: 4
E0: 0 E1: 0 D0: 0 D1: 0 C/I0_3: 0 C/I0_4: 0 C/I0_5: 0 C/I0_6: 0 TIME: 680615 ns FRAME NUMBER: 5
E0: 0 E1: 0 D0: 0 D1: 0 C/I0_3: 0 C/I0_4: 1 C/I0_5: 0 C/I0_6: 1 TIME: 805415 ns FRAME NUMBER: 6
E0: 0 E1: 0 D0: 0 D1: 0 C/I0_3: 0 C/I0_4: 1 C/I0_5: 0 C/I0_6: 1 TIME: 930215 ns FRAME NUMBER: 7
E0: 0 E1: 0 D0: 0 D1: 0 C/I0_3: 0 C/I0_4: 1 C/I0_5: 0 C/I0_6: 1 TIME: 1055015 ns FRAME NUMBER: 8
E0: 0 E1: 0 D0: 0 D1: 0 C/I0_3: 0 C/I0_4: 1 C/I0_5: 0 C/I0_6: 1 TIME: 1179815 ns FRAME NUMBER: 9
E0: 0 E1: 0 D0: 0 D1: 0 C/I0_3: 0 C/I0_4: 1 C/I0_5: 1 C/I0_6: 0 TIME: 1304615 ns FRAME NUMBER: 10
E0: 0 E1: 0 D0: 0 D1: 0 C/I0_3: 0 C/I0_4: 1 C/I0_5: 1 C/I0_6: 0 TIME: 1429415 ns FRAME NUMBER: 11
E0: 0 E1: 0 D0: 0 D1: 0 C/I0_3: 0 C/I0_4: 1 C/I0_5: 1 C/I0_6: 0 TIME: 1554215 ns FRAME NUMBER: 12
E0: 0 E1: 0 D0: 0 D1: 0 C/I0_3: 0 C/I0_4: 1 C/I0_5: 1 C/I0_6: 0 TIME: 1679015 ns FRAME NUMBER: 13
E0: 0 E1: 0 D0: 0 D1: 0 C/I0_3: 0 C/I0_4: 1 C/I0_5: 1 C/I0_6: 0 TIME: 1803815 ns FRAME NUMBER: 14
E0: 0 E1: 0 D0: 0 D1: 0 C/I0_3: 0 C/I0_4: 1 C/I0_5: 1 C/I0_6: 0 TIME: 1928615 ns FRAME NUMBER: 15
E0: 0 E1: 0 D0: 0 D1: 0 C/I0_3: 0 C/I0_4: 1 C/I0_5: 1 C/I0_6: 0 TIME: 2054195 ns FRAME NUMBER: 16
E0: 0 E1: 0 D0: 0 D1: 0 C/I0_3: 0 C/I0_4: 1 C/I0_5: 1 C/I0_6: 0 TIME: 2178345 ns FRAME NUMBER: 17
E0: 0 E1: 0 D0: 0 D1: 0 C/I0_3: 0 C/I0_4: 1 C/I0_5: 0 C/I0_6: 0 TIME: 2303145 ns FRAME NUMBER: 18
E0: 0 E1: 0 D0: 0 D1: 0 C/I0_3: 0 C/I0_4: 1 C/I0_5: 0 C/I0_6: 0 TIME: 2433145 ns FRAME NUMBER: 19
E0: 0 E1: 0 D0: 0 D1: 0 C/I0_3: 0 C/I0_4: 1 C/I0_5: 0 C/I0_6: 0 TIME: 2557945 ns FRAME NUMBER: 20
E0: 0 E1: 0 D0: 0 D1: 0 C/I0_3: 0 C/I0_4: 1 C/I0_5: 0 C/I0_6: 0 TIME: 2682745 ns FRAME NUMBER: 21
E0: 0 E1: 0 D0: 0 D1: 0 C/I0_3: 0 C/I0_4: 0 C/I0_5: 0 C/I0_6: 1 TIME: 2807545 ns FRAME NUMBER: 22
E0: 0 E1: 0 D0: 0 D1: 0 C/I0_3: 1 C/I0_4: 1 C/I0_5: 0 C/I0_6: 1 TIME: 2932345 ns FRAME NUMBER: 23
E0: 0 E1: 1 D0: 0 D1: 1 C/I0_3: 1 C/I0_4: 1 C/I0_5: 0 C/I0_6: 1 TIME: 3057080 ns FRAME NUMBER: 24
E0: 1 E1: 1 D0: 1 D1: 1 C/I0_3: 1 C/I0_4: 1 C/I0_5: 0 C/I0_6: 1 TIME: 3181880 ns FRAME NUMBER: 25
E0: 1 E1: 1 D0: 1 D1: 1 C/I0_3: 1 C/I0_4: 1 C/I0_5: 0 C/I0_6: 1 TIME: 3306615 ns FRAME NUMBER: 26
E0: 1 E1: 1 D0: 1 D1: 1 C/I0_3: 1 C/I0_4: 1 C/I0_5: 0 C/I0_6: 1 TIME: 3431415 ns FRAME NUMBER: 27
E0: 1 E1: 1 D0: 1 D1: 1 C/I0_3: 1 C/I0_4: 1 C/I0_5: 0 C/I0_6: 1 TIME: 3556215 ns FRAME NUMBER: 28
E0: 1 E1: 1 D0: 1 D1: 1 C/I0_3: 1 C/I0_4: 0 C/I0_5: 0 C/I0_6: 1 TIME: 3681015 ns FRAME NUMBER: 29
E0: 1 E1: 1 D0: 1 D1: 1 C/I0_3: 1 C/I0_4: 0 C/I0_5: 0 C/I0_6: 1 TIME: 3805815 ns FRAME NUMBER: 30
E0: 1 E1: 1 D0: 1 D1: 1 C/I0_3: 1 C/I0_4: 0 C/I0_5: 0 C/I0_6: 1 TIME: 3930615 ns FRAME NUMBER: 31
E0: 1 E1: 1 D0: 1 D1: 1 C/I0_3: 1 C/I0_4: 0 C/I0_5: 0 C/I0_6: 1 TIME: 4055415 ns FRAME NUMBER: 32
E0: 1 E1: 1 D0: 1 D1: 1 C/I0_3: 1 C/I0_4: 0 C/I0_5: 0 C/I0_6: 1 TIME: 4180215 ns FRAME NUMBER: 33
E0: 1 E1: 1 D0: 1 D1: 1 C/I0_3: 1 C/I0_4: 0 C/I0_5: 0 C/I0_6: 1 TIME: 4305015 ns FRAME NUMBER: 34
E0: 1 E1: 1 D0: 1 D1: 1 C/I0_3: 1 C/I0_4: 0 C/I0_5: 0 C/I0_6: 1 TIME: 4429815 ns FRAME NUMBER: 35
E0: 1 E1: 1 D0: 1 D1: 1 C/I0_3: 1 C/I0_4: 0 C/I0_5: 0 C/I0_6: 0 TIME: 4554615 ns FRAME NUMBER: 36
E0: 1 E1: 1 D0: 1 D1: 1 C/I0_3: 1 C/I0_4: 1 C/I0_5: 0 C/I0_6: 0 TIME: 4679415 ns FRAME NUMBER: 37
E0: 1 E1: 1 D0: 1 D1: 1 C/I0_3: 1 C/I0_4: 1 C/I0_5: 1 C/I0_6: 1 TIME: 4804215 ns FRAME NUMBER: 38
E0: 1 E1: 1 D0: 1 D1: 1 C/I0_3: 1 C/I0_4: 1 C/I0_5: 1 C/I0_6: 1 TIME: 4929015 ns FRAME NUMBER: 39
E0: 0 E1: 0 D0: 0 D1: 0 C/I0_3: 1 C/I0_4: 1 C/I0_5: 1 C/I0_6: 1 TIME: 5053815 ns FRAME NUMBER: 40
E0: 1 E1: 0 D0: 1 D1: 0 C/I0_3: 1 C/I0_4: 1 C/I0_5: 1 C/I0_6: 1 TIME: 5178615 ns FRAME NUMBER: 41
E0: 1 E1: 0 D0: 1 D1: 0 C/I0_3: 1 C/I0_4: 1 C/I0_5: 1 C/I0_6: 1 TIME: 5303415 ns FRAME NUMBER: 42
E0: 1 E1: 0 D0: 1 D1: 0 C/I0_3: 1 C/I0_4: 1 C/I0_5: 1 C/I0_6: 1 TIME: 5428215 ns FRAME NUMBER: 43
E0: 1 E1: 0 D0: 1 D1: 0 C/I0_3: 1 C/I0_4: 1 C/I0_5: 1 C/I0_6: 1 TIME: 5553015 ns FRAME NUMBER: 44
E0: 1 E1: 0 D0: 1 D1: 0 C/I0_3: 1 C/I0_4: 1 C/I0_5: 1 C/I0_6: 1 TIME: 5677815 ns FRAME NUMBER: 45
E0: 1 E1: 0 D0: 1 D1: 0 C/I0_3: 1 C/I0_4: 1 C/I0_5: 1 C/I0_6: 1 TIME: 5802615 ns FRAME NUMBER: 46
E0: 1 E1: 0 D0: 1 D1: 0 C/I0_3: 1 C/I0_4: 1 C/I0_5: 1 C/I0_6: 1 TIME: 5927415 ns FRAME NUMBER: 47
E0: 1 E1: 0 D0: 1 D1: 0 C/I0_3: 1 C/I0_4: 1 C/I0_5: 1 C/I0_6: 1 TIME: 6052215 ns FRAME NUMBER: 48
E0: 1 E1: 0 D0: 1 D1: 0 C/I0_3: 1 C/I0_4: 1 C/I0_5: 1 C/I0_6: 1 TIME: 6177015 ns FRAME NUMBER: 49
E0: 1 E1: 0 D0: 1 D1: 0 C/I0_3: 1 C/I0_4: 1 C/I0_5: 1 C/I0_6: 1 TIME: 6301815 ns FRAME NUMBER: 50
E0: 1 E1: 0 D0: 1 D1: 0 C/I0_3: 1 C/I0_4: 1 C/I0_5: 1 C/I0_6: 1 TIME: 6426615 ns FRAME NUMBER: 51
E0: 1 E1: 0 D0: 1 D1: 0 C/I0_3: 1 C/I0_4: 1 C/I0_5: 1 C/I0_6: 1 TIME: 6551415 ns FRAME NUMBER: 52
E0: 1 E1: 0 D0: 1 D1: 0 C/I0_3: 1 C/I0_4: 1 C/I0_5: 1 C/I0_6: 1 TIME: 6676215 ns FRAME NUMBER: 53
E0: 0 E1: 1 D0: 1 D1: 0 C/I0_3: 1 C/I0_4: 1 C/I0_5: 1 C/I0_6: 1 TIME: 6801015 ns FRAME NUMBER: 54
E0: 0 E1: 1 D0: 1 D1: 0 C/I0_3: 1 C/I0_4: 1 C/I0_5: 1 C/I0_6: 1 TIME: 6925815 ns FRAME NUMBER: 55
E0: 0 E1: 1 D0: 1 D1: 1 C/I0_3: 1 C/I0_4: 1 C/I0_5: 1 C/I0_6: 1 TIME: 7050615 ns FRAME NUMBER: 56
E0: 0 E1: 1 D0: 1 D1: 1 C/I0_3: 1 C/I0_4: 1 C/I0_5: 1 C/I0_6: 1 TIME: 7175415 ns FRAME NUMBER: 57
E0: 0 E1: 1 D0: 1 D1: 1 C/I0_3: 1 C/I0_4: 1 C/I0_5: 1 C/I0_6: 1 TIME: 7300215 ns FRAME NUMBER: 58
E0: 0 E1: 1 D0: 0 D1: 0 C/I0_3: 1 C/I0_4: 1 C/I0_5: 1 C/I0_6: 1 TIME: 7425015 ns FRAME NUMBER: 59
E0: 0 E1: 1 D0: 0 D1: 0 C/I0_3: 1 C/I0_4: 1 C/I0_5: 1 C/I0_6: 1 TIME: 7549815 ns FRAME NUMBER: 60
E0: 0 E1: 1 D0: 0 D1: 0 C/I0_3: 1 C/I0_4: 1 C/I0_5: 1 C/I0_6: 1 TIME: 7674615 ns FRAME NUMBER: 61
E0: 0 E1: 1 D0: 0 D1: 0 C/I0_3: 1 C/I0_4: 1 C/I0_5: 1 C/I0_6: 1 TIME: 7799415 ns FRAME NUMBER: 62
E0: 0 E1: 1 D0: 0 D1: 0 C/I0_3: 1 C/I0_4: 1 C/I0_5: 1 C/I0_6: 1 TIME: 7924215 ns FRAME NUMBER: 63

Appendix E

VHDL results file – The D bits and E bits result files.

The D Bits file (D_BITS.txt)

0
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
1
1
1
1
1
1
0
0
0
0
0
0
0
0
0
0

1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0

The D bits result file (DBITS_RESULT.txt).

0
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
1
1
1
1
1
0
0
0
0
0
0
0
0
0
0

The E Bits file (E_BITS.txt)

0
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1

1
0
1
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0

The E Bits result file (EBITS_RESULT.txt).

0
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1
0
1